engineering	**Users' Manual for SM 137 - SM 140 motors**	**H5834D0009ING** **02** **05/15/2007**

CANopen Communication Protocol

Publication information

Update List			
Revision	**Added**	**Deleted**	**Changed**
00	Unreleased		
01			Sect. 9.1
02			New Layout

CNI ENGINEERING S.r.l.
No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanic, including photocopying, without the express written permission of **CNI Engineering**.

Manual written by the CNI Engineering Technical Publications Office
H5834D0009ING-02_Cop.fm

PUBLICATION ISSUED BY:

Documentation office

Via Carpanelli, 24
40011 Anzola dell'Emilia (Bo) Italy
Tel. +39 051 6508911
Fax +39 051 6508912
Info@cnicnc.com
www.cnicnc.com

Registered offices

Via dell'Artigianato, 1
48011 Alfonsine (Ra) Italy
Tel. +39 0544 84277
Fax +39 0544 80635
P.I. e C.F. 02248390391

Document Code: **H5834D0009ING**

Document revision: **02**

Document edition: **05/15/2007**

INDEX

1 Introduction to the CANopen communication protocol 1

1.1 "CiA (Can in Automation) specifications 1
1.2 Object Dictionary 1
1.3 DS 301 Protocols 1
1.4 DS301 States 2
1.5 DSP 402 States 3
1.6 Operating modes 5
1.7 Serial communication protocol 5

2 Object Dictionary.....7

2.1 Definition of an object 7
2.1.1 Object description table 7
2.1.2 Object description table 8

3 Object Dictionary DS 301.....11

3.1 Object 1000_h: Device Type 11
3.2 Object 1001_h: Error Register 11
3.3 Object 1002_h:Manufacturer Status Register 12
3.4 Object 1008_h: Manufacturer Device Name 13
3.5 Object 1009_h: Manufacturer Hardware Version 13
3.6 Object $100A_h$: Manufacturer Software Version 14
3.7 Object $100C_h$: Guard Time 15
3.8 Object $100D_h$: Life Time Factor 16
3.9 Object 1010_h: Store Parameters 17
3.10 Object 1011_h: Restore Parameters 18
3.11 Object 1014_h: COB-ID Emergency Message 19
3.12 Object 1400_h: 1st Receive PDO Parameter 20
3.13 Object 1401_h: 2nd Receive PDO Parameter 22
3.14 Object 1600_h: 1st Receive PDO Mapping Parameter 24
3.15 Object 1601_h: 2nd Receive PDO Mapping Parameter 27
3.16 Object 1800_h: 1st Transmit PDO Parameter 30
3.17 Object 1801_h: 2nd Transmit PDO Parameter 32
3.18 Object $1A00_h$: 1st Transmit PDO Mapping Parameter 34
3.19 Object $1A01_h$: 2nd Transmit PDO Mapping Parameter 37

4 Object Dictionary: objects defined by the manufacturer.....41

4.1 Object 2000_h: Proportional Gain of Current Loop 41
4.2 Object 2001_h: Integral Gain of Current Loop 41

4.3 Object 2004_h: Maximum Value of Current Loop Output 42
4.4 Object 2005_h: Proportional Gain of Speed Loop 42
4.5 Object 2006_h: Integral Gain of Speed Loop 43
4.6 Object 2007_h: Percentage Feedforward for Speed Loop 43
4.7 Object 2009_h: Maximum Value of Speed Loop Output 44
4.8 Object $200A_h$: Proportional Gain of Position Loop 44
4.9 Object $200B_h$: Integral Gain of Position Loop 45
4.10 Object $200C_h$: Percentage Feedforward for Position Loop 45
4.11 Object $200E_h$: Maximum Value of Position Loop Output 46
4.12 Object $200F_h$: Maximum Speed Following Error 46
4.13 Object 2010_h: Maximum Duration of Speed Following Error 47
4.14 Object 2011_h: Maximum Torque 47
4.15 Object 2012_h: Timeout for Maximum Torque 48
4.16 Object 2013_h: Bit_A 48
4.17 Object 2014_h: Electric Angle On Zero Index 49
4.18 Object 2015_h: First Component of Speed Loop Feedforward 49
4.19 Object 2016_h: Second Component of Speed Loop Feedforward 50
4.20 Object 2017_h: Third Component of Speed Loop Feedforward 50
4.21 Object 2018_h: Fourth Component of Speed Loop Feedforward 51
4.22 Object 3000_h: Special Commands 51
4.23 Object 3010_h: Sampling Variables 54
4.24 Object $5f00_h$: Reserved 55
4.25 Table Bit_A 56

5 Objects Dictionary: DSP402 57

5.1 Object 6040_h: Controlword 57
5.2 Object 6041_h: Statusword 58
5.2.1 Controlword and Statusword in 'Profile Position Mode' 60
5.2.2 Controlword and Statusword in 'Profile Velocity Mode' 66
5.2.3 Controlword and Statusword in 'Homing Mode' 67
5.3 Object 6060_h: Modes of operation 69
5.4 Object 6061_h: Modes of operation display 71
5.5 Object 6064_h: Position Actual Value 72
5.6 Object 6065_h: Following Error Window 72
5.7 Object 6066_h: Following Error Timeout 73
5.8 Object 6067_h: Position Window 73
5.9 Object 6068_h: Position Window Time 74
5.10 Object $606B_h$: Velocity Demand Value 75
5.11 Object $606C_h$: Velocity Actual Value 75
5.12 Object $607A_h$: Target Position 76
5.13 Object $607C_h$: Homing Offset 76
5.14 Object $607D_h$: Software Position Limit 77
5.15 Object 6081_h: Profile Velocity 78
5.16 Object 6083_h: Profile Acceleration 78
5.17 Object 6085_h: Quick Stop Deceleration 79
5.18 Object 6098_h: Homing Method 79
5.19 Object 6099_h: Homing Speed Value 80
5.20 Object $60FF_h$: Target Velocity 80

6 Objects that are not implemented 81

6.1 Object 6007_h: Abort Connection Option Code 81
6.2 Object $605A_h$: Quick Stop Option Code 82
6.3 Object $605C_h$: Disable Operation Option Code 83
6.4 Object $605B_h$: Shutdown Option Code 84
6.5 Object $605D_h$: Halt Option Code 85
6.6 Object $605E_h$: Fault Reaction Option Code 86

7 Emergency Messages 87

8 SDO Interrupt Codes 89

9 Dip-switch configuration 91

9.1 Baudrate 91
9.2 Restoring Default parameters 91
9.3 Node number 92
9.4 Meaning of the green LED 92

1 Introduction to the CANopen communication protocol

This section summarises the basic concepts behind the CANopen communication protocol. For a more thorough discussion, please refer to the following documents:

- "CANopen Application Layer and Communication Profile - CiA Draft Standard 301 - Version 4.02".

1.1 "CiA (Can in Automation) specifications

In the attempt to standardise the various devices that can be used on a CAN bus, the CiA has issued a series of specifications with which a device must comply in order to be considered "CANopen". These specifications define the objects that must/may be present in the Object Dictionary and the various communication protocols implemented. The two reference specifications used in this manual are "DS 301" which refers to a generic device, and "DSP 402" relating to servomotors.
In the following we will first look at the contents of the DS 301 specifications, as these are general and relate to all CANopen devices. Subsequently we will also discuss the specification DSP 402 and thus take a more detailed look at how a CANopen servomotor operates.

1.2 Object Dictionary

Each CANopen device has an “object dictionary” containing all the device parameters. By reading and writing these parameters it is possible to send commands to the CANopen device and monitor its state. Each object is defined by an index and a subindex which identify that object. For example: the real position of the motor is identified by the object with index 0x6064 and subindex 0x00.

1.3 DS 301 Protocols

The types of protocol defined in the DS 301 specifications are as follows:

- SDO protocol. This allows reading and writing of the device objects, whatever their length and type. It involves the exchange of index, subindex and length, as well as the value of the object it accesses. The protocol can be divided into write SDO (download) and read SDO (upload).
- PDO protocol. This allows efficient reading and writing only of the so-called “mappable” objects. The two communicating devices must first define which objects are to be exchanged using this protocol. When they switch to “operational” the devices will start to communicate the parameter values only (without either index or subindex). Each PDO can contain a maximum of 8 byte. PDO objects can be divided into received PDO and transmitted PDO.
- Sync protocol. Used to synchronise various devices to carry out the PDO data. It is essential, for example, if 2 or more axes are to be interpolated simultaneously in a synchronous manner.
- Emergency protocol. Dedicated to communication of device malfunctions.
- NMT network management (Network ManagemenT) protocol. Used to initialise, monitor, reset or stop any device in the network.
  In particular one of the services that this protocol makes available is “error checking” using so-called "Node Guarding": the CAN bus manager (also known as the master) sends a package to each device (slave) at fixed times. If the slave does not respond or responds in the wrong manner the master will detect this, or if the slave does not receive the package within the set time it will be switched to a state of safety.

Another service available is that of "bootup" which involves sending an NMT message when the CANopen device is switched on.

1.4 DS301 States

A generic CANopen device has various states with regard to communication. The various states differ from each other above all in the communication protocols they support. Figure 1 and table 1 represent the automatic device defined in the specification DS 301:

Figure 1 DS 301 States

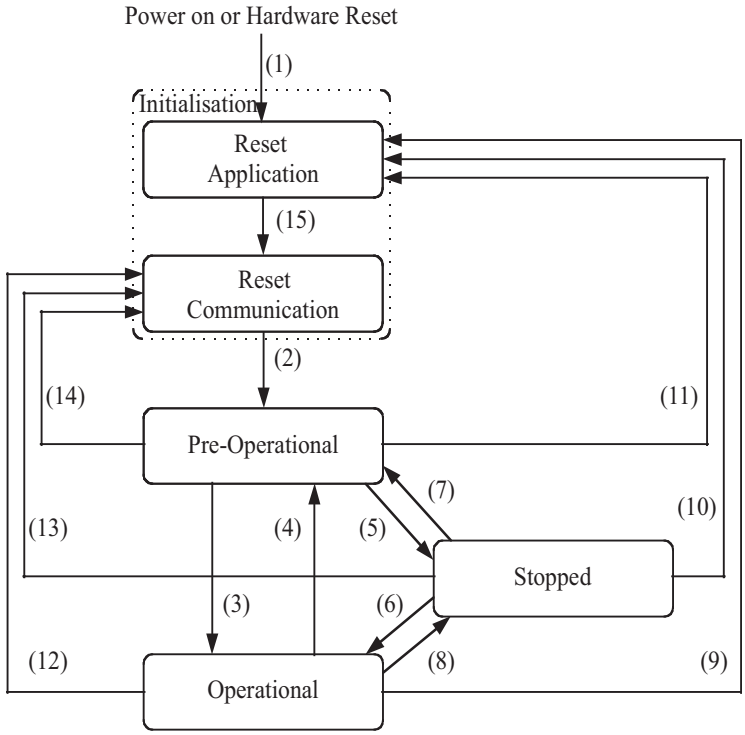


Table 1 DS301 Transitions

Transition	Event involving the transition
(1)	On switching on, initialisation starts automatically.
(2)	On completion of initialisation, the device automatically enters Pre-Operational state.
(3),(6)	A "Start_Remote_Node" is received from the master (via the NMT protocol).
(4),(7)	An "Enter_Pre-Operational_State" is received from the master (via the NMT protocol).
(5),(8)	A "Stop_Remote_Node" is received from the master.
(9),(10),(11)	A "Reset_Node" is received from the master.
(12),(13),(14)	A "Reset_Communication" is received from the master.
(15)	As soon as the "Reset_Application" phase has terminated, "Reset_Communication" starts automatically.

The protocols supported by the various states are shown in table 2:

Table 2 Protocols supported

State	Meaning	Protocols					
		SDO	PDO	SYNC	EMGCY	NMT	BOOTUP
Reset Application	The hardware is initialised and the typical device parameters are reloaded.						X
Reset Communication	The CANopen parameters are reloaded from flash.						X
Pre-Operational	The device is operational.	X		X	X	X	
Operational	The device is operational.	X	X	X	X	X	
Stopped	The device is in safety mode.					X	

1.5 DSP 402 States

As already mentioned, the specification DSP 402 defines the "standard behaviour" of a CANopen servomotor. The states of the motor from a drive point of view are defined in the specification. The transitions automatic device is outlined in figure 2:

Figure 2 DSP 402 States

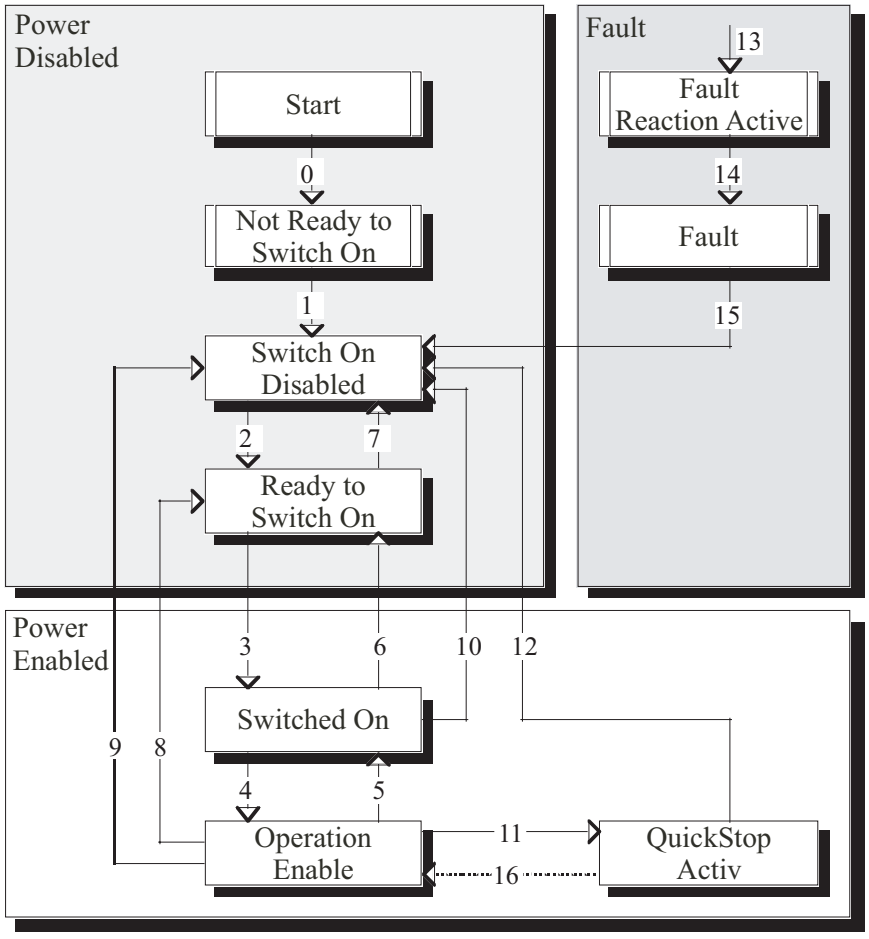


Table 3 DSP 402 States

State	Description
Start	This is the initial state on start-up.
Not Ready to Switch On	During this phase the servomotor performs diagnostic tests and initialisation. If the brake is present it is enabled. The motor is not engaged.
Switch On Disabled	Initialisation has been completed. The parameters have been loaded and can be changed. The motor is not engaged.
Ready to Switch On	The parameters can be changed. The motor is not engaged.
Switched On	The power part of the motor is active. The motor is engaged. The parameters can be changed. There are no malfunctions.
Operation Enable	The servomotor functions are active. The motor is engaged. There are no malfunctions.
Quick Stop Active	The motor is stopping or is already stopped. The motor is engaged. If the motor is stopped the parameters can be changed.
Fault Reaction Active	The motor is switching to safety mode due to a malfunction.
Fault	The motor parameters can be changed. The motor is not engaged.

Transitions from one state to another are determined either by motor errors or by commands sent by the master. These commands are sent using the object "Controlword" defined by DSP402. These transitions are listed in table 4:

Table 4 Commands to determine the changes in state

Transition	Command	Internal state
0		The motor is turned on.
1		The self-diagnosis and initialisation procedures have been completed.
2	Shutdown	
3	Switch On	
4	Enable Operation	
5	Disable Operation	
6	Shutdown	
7	Quick Stop or Disable Voltage	
8	Shutdown	
9	Disable Voltage	
10	Quick Stop or Disable Voltage	
11	Quick Stop	
12	Disable Voltage	
13		An error has occurred.
14		The motor has switched to safety mode.
15	Fault Reset	
16	Operation Enable	

1.6 Operating modes

The servomotor has various modes of operation, which are active during "operation enable" state and allow various activities to be carried out:

1. Resetting of the motor positions (homing mode).
2. Setting of the motor speed (profile velocity mode).
3. Setting of the motor position and speed (profile position mode).

1.7 Serial communication protocol

As regards use of Smart Motor SM137 and SM140 with the serial communication protocol, please refer to the “User Manual for SM137-SM140 motors: Serial communication protocol” distributed by CNI Engineering S.r.l.

2 Object Dictionary

This section describes the Object Dictionary, that is to say the series of objects defined within the Smart Motor that allows commands to be send and/or dimensions such as the position and the speed to be monitored.

2.1 Definition of an object

Each object is defined using the following tables:

- Object description table.
- Input description table.
- Format description table.
- Data description table.

2.1.1 Object description table

Description of object	Index	Numerical identification for the object.
	Object name	Symbolic name of the object.
	Object code	Structure of the object.
	Data type	Type of object.
	Reference	Reference to the CiA profile.

Index

The index of an object is defined in hexadecimal form; this can be identified by the fact that there is a lower case letter ‘h’ at the foot of the index for the object in question.
The following types of index are used:

- "1000_h - $1FFF_h$ Objects with communication profile *CiA DS-301 V4.02*
- "2000_h - $5FFF_h$ Objects defined by the manufacturer.
- "6000_h - $9FFF_h$ Objects with the device profile *CiA DSP-402 V2.0*

Object Name

Indicates the symbolic name of the object.

Object Code

One of the following object structures (Object Codes) will be assigned to all objects:

- VAR: Single value, for example of the type Integer8, Unsigned32 etc.
- ARRAY: A set of data in which all the elements have the same data type. The subindex 00_h determines the number of elements.
- RECORD: A set of data made up of elements of different type. The subindex 00_h determines the number of elements.

Data type

The data types that an object can have are the following:

Name	Description	Interval	Bytes
Integer8	8 bit values	-128,...,+127	1
Integer16	16 bit values	-32768,...,+32767	2
Integer32	32 bit values	-2147483648,...,+2147483647	4
Unsigned8	8 bit values	0,...,255	1
Unsigned16	16 bit values	0,...,65535	2
Unsigned32	32 bit values	0,...,4294967295	4
Visible string	ASCII symbols	20_h,...,$7E_h$	≤15
String of octets	ASCII symbols	00_h,...,FF_h	≤16

CiA Reference

The Reference row in the definition of an object contains a reference to the CAN profile definition used in the CiA (CAN in Automation) manual.

2.1.2 Object description table

Description of elements	Subindex	Sequential number identifying the object from other objects with the same index.
	Description	Indicates the name of the parameter.
	Access	Indicates the object's access attribute.
	PDO mapping	Determines whether or not the object can be mapped in a PDO.
	Unit	Indicates the unit of measurement for the object.
	Interval of values	Indicates the interval of values for the object.
	Preset value	Indicates the preset value of the object.
	Recordable	Indicates whether or not the object can be recorded in a permanent manner.

Subindex

The subindex of an object comprises a hexadecimal number identified by a lower case letter 'h' at the foot of the object's subindex.
In the case of single objects (VAR), only the subindex 00_h will be assigned. In the case of intervals of values (ARRAY or RECORD), the value of subindex 00_h determines the size of the group of values.

Description

Indicates the name of the parameter in question.

Access

An access attribute will be assigned for each object:

Value	Description
rw	The value of the object is both readable and writable.
ro	The value of the object is read-only.
wo	The value of the object is write-only.
const	Read-only access, the value is constant.

PDO mapping

The “PDO mapping” line of an object in input indicates whether or not the parameter can be mapped within a PDO (Process Data Object).

Unit of measurement

The physical unit for parameters must be defined explicitly for each object (Unit).

- The positions are expressed in counts [cnt]
- The speeds are expressed in rpm [rpm]
- The accelerations are expressed in revs per second squared divided by 10000 [$r/s^2/10000$]
- The currents are expressed in Ampere multiplied by 100 [Ax100]
- PI regulator gains are pure numbers multiplied by 100 [x100]

Interval of values

The interval of values for an object is restricted by the type of data, unless the interval of values allowed is not stated explicitly.

Preset value

All devices are supplied with preset parameter values (values on delivery or factory values). The operator can adjust modifiable parameters for his application, and then record them in the non volatile memory.

Recordable

The “Recordable” line indicates whether or not the object can be recorded in the non volatile memory.

3 Object Dictionary DS 301

3.1 Object 1000_h: Device Type

Object 1000_h describes the Device Type and the device profile applied.

.

Description of object	Index	1000_h
	Object name	Device Type
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DS-301 V4.04, page 86; CiA DSP-402 V2.0, page 24
Description of elements	Subindex	00_h
	Description	Device type
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	00020192_h
	Recordable	No
Description of format	Bit 31 - 24	Not used
	Bit 23 - 16	Inverter type (Bit 17 = 1: servo-driven)
	Bit 15 - 0	Device CiA profile (0192_h=402)

3.2 Object 1001_h: Error Register

Object 1001_h is an error register for the device.

.

Description of object	Index	1001_h
	Object name	Error Register
	Object code	VAR
	Data type	Unsigned8
	Reference	CiA DS-301 V4.02, page 87; CiA DSP-402 V2.0, page 24

Description of elements	Subindex	00_h
	Description	Error register
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	0
	Recordable	Yes
Description of format	Bit 7	Specific to manufacturer.
	Bit 6	Reserved (always 0).
	Bit 5	Specific to device CiA profile.
	Bit 4	Communication error (overrun, error state).
	Bit 3	Temperature.
	Bit 2	Power.
	Bit 1	Current.
	Bit 0	Generic error.

3.3 Object 1002_h:Manufacturer Status Register

Object 1002_h is a status register for element customised by the manufacturer.

.

Description of object	Index	1002_h
	Object name	Manufacturer Status Register
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DS-301 V4.0, pages 9-65
Description of elements	Subindex	00_h
	Description	Manufacturer status register
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0
	Recordable	Only bits from 15 to 0

3.4 Object 1008_h: Manufacturer Device Name

The object with index 1008_h contains the device name assigned by the manufacturer.

.

Description of object	Index	1008_h
	Object name	Manufacturer Device Name
	Object code	VAR
	Data type	Visible string
	Reference	CiA DS-301 V4.02, page 91
Description of elements	Subindex	00_h
	Description	Device name
	Access	const
	PDO mapping	No
	Unit	-
	Interval of values	Visible string (≤15 characters)
	Preset value	SM137
	Recordable	No

3.5 Object 1009_h: Manufacturer Hardware Version

The object with index 1009_h contains the description of the hardware version assigned by the manufacturer.

.

Description of object	Index	1009_h
	Object name	Manufacturer Hardware Version
	Object code	VAR
	Data type	Visible String
	Reference	CiA DS-301 V4.02, page 91
Description of elements	Subindex	00_h
	Description	Manufacturer hardware version
	Access	const
	PDO mapping	No
	Unit	-
	Interval of values	Visible string (≤15 characters)
	Preset value	P137C
	Recordable	No

3.6 Object $100A_h$: Manufacturer Software Version

The object with index $100A_h$ contains the description of the software version loaded on the device.

Description of object	Index	$100A_h$
	Object name	Manufacturer Software Version
	Object code	VAR
	Data type	Visible String
	Reference	CiA DS-301 V4.02, page 91
Description of elements	Subindex	00_h
	Description	Software version
	Access	const
	PDO mapping	No
	Unit	-
	Interval of values	Visible string (≤15 characters)
	Preset value	Currently 00120 or 00123
	Recordable	No

3.7 Object $100C_h$: Guard Time

The objects with indexes $100C_h$ and $100D_h$ include the Guard Time in milliseconds and the Life Time Factor. The Life Time Factor multiplied by the Guard Time gives the maximum interval for the Life Guarding Protocol: if, within this time, the peripheral device (NMT-Slave) does not receive Node Guarding packages from the Master, the peripheral device will revert to safety mode. This has a value of 0 if Node Guarding (surveillance of the CANopen node) is disabled.
By means of the Node Guarding protocol, an NMT-Master supervises communication with the peripheral devices (NMT-Slaves). The Guard time is indicated in milliseconds.

Description of object	Index	$100C_h$
	Object name	Guard Time
	Object code	VAR
	Data type	Unsigned16
	Reference	CiA DS-301 V4.02, page 92
Description of elements	Subindex	00_h
	Description	Peripheral device watch-dog
	Access	rw (read/write)
	PDO mapping	No
	Unit	milliseconds
	Interval of values	Unsigned16
	Preset value	0
	Recordable	Yes

3.8 Object $100D_h$: Life Time Factor

The object with index $100D_h$ regulates the Life Time for Life-Guarding.
The Life Time Factor multiplied by the Cycle Time of the Guard Time (object $100C_h$ peripheral device watch-dog) gives the Life Time. The Life Time Factor is set to 0 if it is not used
The reaction of the SmartMotor to a loss of connection with the NMT-Master (Life Guarding Event), can be regulated using object 6007_h (Abort Connection Option Code)*.

Description of object	Index	$100D_h$
	Object name	Life Time Factor
	Object code	VAR
	Data type	Unsigned8
	Reference	CiA DS-301 V4.02, page 92
Description of elements	Subindex	00_h
	Description	Life time factor
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	0
	Recordable	Yes

* Object 6007_h (Abort Connection Option Code) is currently not implemented. The behaviour of the motor is the same as if the object had a value of 1: a malfunction that switches the SM to a state of Fault.

3.9 Object 1010_h: Store Parameters

The object with index 1010_h allows the parameters to be stored in a non volatile memory. To avoid parameter storage errors, storage only takes place when a specific code is entered in the appropriate subindex. This code is "save". By writing the expression 'save' in the subindex, the current parameter values are saved to the non volatile memory and are therefore available when the device is turned on again (Power-On Defaults).
WARNING: The parameters are saved to the processor's Flash memory! The number of save operations that can be performed is therefore limited..

Description of object	Index	1010_h
	Object name	Store Parameters
	Object code	ARRAY
	Data type	Unsigned32
	Reference	CiA DS-301 V4.02, page 92
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1 - 255
	Preset value	1
	Recordable	No
	Subindex	01_h
	Description	Save all parameters
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	1
	Recordable	No
Description of format Subindex 01_h(Read access)	Bit 31 - 2	reserved
	Bit 1	0= device does not save the parameters independently
		1= device saves the parameters independently
	Bit 0	0= device does not save the parameters on command
		1= device saves the parameters on command
Description of format Subindex 01_h(Write access)	Bit 31 - 24	65_h = 'e' (ASCII characters, ISO 8859)
	Bit 23 - 16	76_h = 'v' (ASCII characters, ISO 8859)
	Bit 15 - 8	61_h = 'a' (ASCII characters, ISO 8859)
	Bit 7 - 0	73_h = 's' (ASCII characters, ISO 8859)

3.10 Object 1011_h: Restore Parameters

The object with index 1011_h is used to restore the parameter values preset by the manufacturer according to the communication or profile of the device. During read access, the device provides information on its ability to regenerate these values.
By writing the instruction ‘load’ in the subindex, the factory settings for the corresponding parameters will be restored, and the factory settings will be restored as the current parameter values after the command “NMT Reset Node” has been performed or after the peripheral device has been turned off and then on again. These parameters can be saved in the permanent memory, and they will be available as Power-On Defaults.

Description of object	Index	1011_h
	Object name	Restore Default Parameters
	Object code	ARRAY
	Data type	Unsigned32
	Reference	CiA DS-301 V4.02, pages 9-72
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1
	Preset value	1
	Recordable	No
	Subindex	01_h
	Description	Restore all preset parameter values.
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	1
	Recordable	No
Description of format Subindex 01_h (Read Access)	Bit 31 - 1	reserved
	Bit 0	0= the device does not restore the parameters
		1= the device restores the parameters
Description of format Subindex 01_h (Write Access)	Bit 31 - 24	64_h = 'd' (ASCII characters, ISO 8859)
	Bit 23 - 16	61_h = 'a' (ASCII characters, ISO 8859)
	Bit 15 - 8	$6F_h$ = 'o' (ASCII characters, ISO 8859)
	Bit 7 - 0	$6C_h$ = 'l' (ASCII characters, ISO 8859)

3.11 Object 1014_h: COB-ID Emergency Message

The object 1014_h defines the COB-ID for the object Emergency (EMCY).

Description of object	Index	1014_h
	Object name	COB-ID Emergency Message
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DS-301 V4.02, page 98
Description of elements	Subindex	00_h
	Description	COB-ID of the emergency message (EMCY)
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	80_h + Node-ID
	Recordable	Yes
Description of format	Bit 31	0 = EMCY exists / is valid 1 = EMCY does not exist / is not valid
	Bit 30	Reserved (always 0)
	Bit 29	0 = 11-bit identifiers (CAN 2.0A) 1 = 29-bit identifiers (CAN2.0B)
	Bit 28 - 11	if bit 29=1, identify bits 28-11 of the 29-bit
	Bit 10-0	Bits 10-0 of the COB-ID

Using the object Emergency, the errors are communicated to the Master at the time they occur. An emergency message is made up of 8 bytes and has the following structure:

Byte 0 - 1	Emergency Error Code
Byte 2	Object 1001_h: Error Register
Byte 3 - 7	Field containing the manufacturer-specific errors

The emergency error codes used are the ones described in the chapter “Emergency Messages” provided herein.

3.12 Object 1400_h: 1st Receive PDO Parameter

The object 1400_h allows customisation of the communication parameters for the first receive PDO (RPDO1).

Description of object	Index	1400_h
	Object name	1st Receive PDO Parameter
	Object code	RECORD
	Data type	PDO CommonPar
	Reference	CiA DS-301 V4.02, page 107; CiA DS-402 V2.0, page 25
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 2 - 5
	Preset value	2
	Recordable	No
	Subindex	01_h
	Description	COB-ID used by the PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	40000200_h + Node-ID
	Recordable	Yes
	Subindex	02_h
	Description	Transmission type
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	255
	Recordable	Yes

Description of format Subindex 01_h	Bit 31	0 = PDO valid 1 = PDO not valid
	Bit 30	0 = RTR allowed on this PDO 1 = RTR not allowed on this PDO
	Bit 29	0 = 11-bit identifiers (CAN 2.0A) 1 = 29-bit identifiers (CAN 2.0B)
	Bit 28 - 11	if bit 29=1, identify bits 28-11 of the 29-bit
	Bit 10 - 0	Bits 10-0 of the COB-ID
Description of data Subindex 02_h	0	synchronous: RPDO1 is synchronised by the next SYNC
	1,...,240	synchronous: same function as value 0
	241,...,251	reserved
	252	not allowed in receive PDOs
	253	not allowed in receive PDOs
	254	asynchronous: same value as 255
	255	asynchronous: RPDO1 is immediately active (straight after receiving) (preset)

Receive PDOs are only processed in a state of NMT OPERATIONAL.
The PDO communication parameters can only be changed in a state of NMT PRE-OPERATIONAL.

3.13 Object 1401_h: 2nd Receive PDO Parameter

The object 1401_h allows customisation of the communication parameters for the second receive PDO (RPDO2) (see also paragraph 3.12 object 1400_h: 1st Receive PDO Parameter).

Description of object	Index	1401_h
	Object name	2nd Receive PDO Parameter
	Object code	RECORD
	Data type	PDO CommonPar
	Reference	CiA DS-301 V4.02, page 107; CiA DS-402 V2.0, page 25
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read-only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 2 - 5
	Preset value	2
	Recordable	No
	Subindex	01_h
	Description	COB-ID used by the PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	40000300_h + Node-ID
	Recordable	Yes
	Subindex	02_h
	Description	Transmission type
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	255
	Recordable	Yes

Description of format Subindex 01_h	Bit 31	0 = PDO valid 1 = PDO not valid
	Bit 30	0 = RTR allowed on this PDO 1 = RTR not allowed on this PDO
	Bit 29	0 = 11-bit identifiers (CAN 2.0A) 1 = 29-bit identifiers (CAN 2.0B)
	Bit 28 - 11	if bit 29=1, identify bits 28-11 of the 29-bit
	Bit 10 - 0	Bits 10-0 of the COB-ID
Description of data Subindex 02_h	0	synchronous: RPDO2 is synchronised by the next SYNC
	1,...,240	synchronous: same function as value 0
	241,...,251	reserved
	252	not allowed in receive PDOs
	253	not allowed in receive PDOs
	254	asynchronous: same value as 255
	255	asynchronous: RPDO2 is immediately active (straight after receiving) (preset)

3.14 Object 1600_h: 1st Receive PDO Mapping Parameter

The object with index 1600_h contains the mapping for the PDOs that the device is enabled to receive.

Description of object	Index	1600_h
	Object name	1st Receive PDO Mapping Parameter
	Object code	RECORD
	Data type	PDO mapping
	Reference	CiA DS-301 V4.02, page 109; CiA DS-402 V2.0, page 25
Description of elements	Subindex	00_h
	Description	Number of mapped objects requested in PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1 - 8
	Preset value	1
	Recordable	Yes
	Subindex	01_h
	Description	First object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	60400010_h
	Recordable	Yes
	Subindex	02_h
	Description	Second object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes

Description of elements	Subindex	03_h
	Description	Third object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	04_h
	Description	Fourth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	05_h
	Description	Fifth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	06_h
	Description	Sixth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	07_h
	Description	Seventh object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32

Description of elements	Subindex	08_h
	Description	Eighth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
Description of format Subindex 01_h - 08_h	Bit 31 - 16	index of the object to be mapped (16 bit)
	Bit 16 - 8	subindex of the object to be mapped (8 bit)
	Bit 7 - 0	length of the object in bits (8 bit)

PDO mapping is dynamic: it is possible to define which objects are mapped within the PDO. This mapping can only be carried out in Pre-operational state.
For example, if in the 1st PDO you wish to receive not only the Controlword (6040_h), but also the objects Mode of Operation (6060_h) and Target position ($607A_h$) it will be necessary to:

read the value of the object 1400_h;

1. in the value read set bit 31 to 1 so as to invalidate the 1st PDO in receive. Send the value calculated to object 1400_h;
2. in object 1600_h subindex 00_h write the value 0 to indicate that no object is mapped;
3. in object 1600_h subindex 01_h write the value 60400010_h to indicate that the first mapped object is the Controlword (address 6040_h subindex 00_h) with a length of 16 bit (2 byte);
4. in object 1600_h subindex 02_h write the value 60600008_h to indicate that the second mapped object is the Mode of Operation (address 6060_h subindex 00_h) with a length of 8 bit (1 byte);
5. in object 1600_h subindex 03_h write the value $607A0020_h$ to indicate that the third mapped object is the Target Position (address $607A_h$ subindex 00_h) with a length of 32 bit (4 byte);
6. Enable the PDO again by setting bit 31 in object 1400_h to 0.

3.15 Object 1601_h: 2nd Receive PDO Mapping Parameter

The object with index 1601_h contains the mapping for the PDO that the device is enabled to receive (see also paragraph 3.14 object 1600_h: 1st Receive PDO Mapping Parameter).

Description of object	Index	1601_h
	Object name	2nd Receive PDO Mapping
	Object code	RECORD
	Data type	PDO mapping
	Reference	CiA DS-301 V4.02, page 109; CiA DS-402 V2.0, page 25

Description of elements	Subindex	00_h
	Description	Number of mapped objects requested in the PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1 - 8
	Preset value	1
	Recordable	Yes
	Subindex	01_h
	Description	First object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	60400010_h
	Recordable	Yes
	Subindex	02_h
	Description	Second object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	6060008_h
	Recordable	Yes
	Subindex	03_h
	Description	Third object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	04_h
	Description	Fourth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32

Description of elements	Subindex	05_h
	Description	Fifth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	06_h
	Description	Sixth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	07_h
	Description	Seventh object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	08_h
	Description	Eighth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
Description of format Subindex 01_h - 08_h	Bit 31 - 16	index of the object to be mapped (16 bit)
	Bit 16 - 8	subindex of the object to be mapped (8 bit)
	Bit 7 - 0	length of the object in bits (8 bit)

PDO mapping is dynamic: it is possible to define which objects are mapped within the PDO. This mapping can only be carried out in Pre-operational state.

3.16 Object 1800_h: 1st Transmit PDO Parameter

The object 1800_h is used to customise the communication parameters for the first transmit PDO.

Description of object	Index	1800_h
	Object name	1st Transmit PDO Parameter
	Object code	RECORD
	Data type	PDO CommPar
	Reference	CiA DS-301 V4.02, page 111; CiA DS-402 V2.0, page 29
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read/only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 2 - 5
	Preset value	5
	Recordable	No
	Subindex	01_h
	Description	COB-ID used by the PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	40000180_h + Node-ID
	Recordable	Yes
	Subindex	02_h
	Description	Transmission type
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	255
	Recordable	Yes

Description of elements	Subindex	03_h
	Description	PDO inhibition time
	Access	rw (read/write)
	PDO mapping	No
	Unit	multiples of 100 microseconds
	Interval of values	Unsigned16
	Preset value	0
	Recordable	Yes
	Subindex	05_h
	Description	Event time interval
	Access	rw (read/write)
	PDO mapping	No
	Unit	milliseconds
	Interval of values	Unsigned16
	Preset value	0
	Recordable	Yes
Description of format Subindex 01_h	Bit 31	0 = PDO valid 1 = PDO not valid
	Bit 30	0 = RTR allowed on this PDO 1 = RTR not allowed on this PDO
	Bit 29	0 = 11-bit identifiers (CAN 2.0A) 1 = 29-bit identifiers (CAN 2.0B)
	Bit 28 - 11	if bit 29=1, identify bits 28-11 of the 29-bit
	Bit 10 - 0	Bits 10-0 of the COB-ID
Description of data Subindex 02_h	0	synchronous acyclic, data updated on previous SYNC
	1,...,240	synchronous cyclic, data updated on previous SYNC
	241,...,251	reserved
	252	synchronous on request, data is updated on previous SYNC
	253	asynchronous on request, data is updated immediately on SYNC
	254	asynchronous: identical to value 255
	255	asynchronous transmitted simultaneously to variation of one of the mapped objects or on expiry of the Event Timer (preset)

The parameter Inhibit Time represents the minimum time (in steps of 100 microseconds) between two successive PDO transmits.
The parameter Event Timer involves a fixed time PDO transmit (prescribed by the parameter in milliseconds) even if the values of the mapped objects have not changed.
The PDO transmit will only take place in NMT OPERATIONAL state. The PDO communication parameters can only be changed in NMT PRE-OPERATIONAL state.

3.17 Object 1801_h: 2nd Transmit PDO Parameter

Object 1801_h is used to customise the communication parameters for the second transmit PDO (see also paragraph 3.16 object 1800_h: 1st Transmit PDO Parameter).

Description of object	Index	1801_h
	Object name	2nd Transmit PDO Parameter
	Object code	RECORD
	Data type	PDO CommPar
	Reference	CiA DS-301 V4.02, page 111; CiA DS-402 V2.0, page 29
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read/only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 2 - 5
	Preset value	5
	Recordable	No
	Subindex	01_h
	Description	COB-ID used by the PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	40000280_h + Node-ID
	Recordable	Yes
	Subindex	02_h
	Description	Transmission type
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8
	Preset value	255
	Recordable	Yes

Description of elements	Subindex	03_h
	Description	PDO inhibition time
	Access	rw (read/write)
	PDO mapping	No
	Unit	multiples of 100 microseconds
	Interval of values	Unsigned16
	Preset value	0
	Recordable	Yes
	Subindex	05_h
	Description	Event time interval
	Access	rw (read/write)
	PDO mapping	No
	Unit	milliseconds
	Interval of values	Unsigned16
	Preset value	0
	Recordable	Yes
Description of format Subindex 01_h	Bit 31	0 = PDO valid 1 = PDO not valid
	Bit 30	0 = RTR allowed on this PDO 1 = RTR not allowed on this PDO
	Bit 29	0 = 11-bit identifiers (CAN 2.0A) 1 = 29-bit identifiers (CAN 2.0B)
	Bit 28 - 11	if bit 29=1, identify bits 28-11 of the 29-bit
	Bit 10 - 0	Bits 10-0 of the COB-ID
Description of data Subindex 02_h	0	synchronous acyclic, data updated on previous SYNC
	1,...,240	synchronous cyclic, data updated on previous SYNC
	241,...,251	reserved
	252	synchronous on request, data is updated on previous SYNC
	253	asynchronous on request, data is updated immediately on SYNC
	254	asynchronous: identical to value 255
	255	asynchronous transmitted simultaneously to variation of one of the mapped objects or on expiry of the Event Timer (preset)

The parameter Inhibit Time represents the minimum time (in steps of 100 microseconds) between two successive PDO transmits.
The parameter Event Timer involves a fixed time PDO transmit (prescribed by the parameter in milliseconds) even if the values of the mapped objects have not changed. The PDO transmit will only take place in NMT OPERATIONAL state. The PDO communication parameters can only be changed in NMT PRE-OPERATIONAL state.

3.18 Object $1A00_h$: 1st Transmit PDO Mapping Parameter

The object $1A00_h$ contains the mapping for the first transmit PDO.

Description of object	Index	$1A00_h$
	Object name	1st Transmit PDO Mapping Parameter
	Object code	RECORD
	Data type	PDO mapping
	Reference	CiA DS-301 V4.02, page 112; CiA DS-402 V2.0, page 29
Description of elements	Subindex	00_h
	Description	Number of objects mapped in the first transmit PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1 - 8
	Preset value	1
	Recordable	Yes
	Subindex	01_h
	Description	First object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	60410010_h
	Recordable	Yes
	Subindex	02_h
	Description	Second object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes

Description of elements	Subindex	03_h
	Description	Third object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	04_h
	Description	Fourth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	05_h
	Description	Fifth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	06_h
	Description	Sixth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	07_h
	Description	Seventh object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32

Description of elements	Subindex	08_h
	Description	Eighth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
Description of format Subindex 01_h - 08_h	Bit 31 - 16	index (16 bit)
	Bit 16 - 8	subindex (8 bit)
	Bit 7 - 0	length (8 bit)

PDO mapping is dynamic: it is possible to define which objects are mapped within the PDO (see also paragraph 3.12 object 1400_h: 1st Receive PDO Parameter). This mapping can only be carried out in Pre-operational state.

3.19 Object $1A01_h$: 2nd Transmit PDO Mapping Parameter

The object $1A01_h$ contains the mapping for the second transmit PDO.

Description of object	Index	$1A01_h$
	Object name	2nd Transmit PDO Mapping Parameter
	Object code	RECORD
	Data type	PDO mapping
	Reference	CiA DS-301 V4.02, page 112; CiA DS-402 V2.0, page 29
Description of elements	Subindex	00_h
	Description	Number of objects mapped in the second transmit PDO
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 1 - 8
	Preset value	2
	Recordable	Yes
	Subindex	01_h
	Description	First object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	60410010_h
	Recordable	Yes

Description of elements	Subindex	02_h
	Description	Secondobject mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	6061008_h
	Recordable	Yes
	Subindex	03_h
	Description	Third object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	04_h
	Description	Fourth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	05_h
	Description	Fifth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	06_h
	Description	Sixth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32

Description of elements	Subindex	07_h
	Description	Seventh object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
	Subindex	08_h
	Description	Eighth object mapped
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned32
	Preset value	0_h
	Recordable	Yes
Description of format Subindex 01_h - 08_h	Bit 31 - 16	index of the object to be mapped (16 bit)
	Bit 16 - 8	subindex of the object to be mapped (8 bit)
	Bit 7 - 0	length of the object in bits (8 bit)

PDO mapping is dynamic: it is possible to define which objects are mapped within the PDO (see also paragraph 3.12 object 1400_h: 1st Receive PDO Parameter). This mapping can only be carried out in Pre-operational state.

4 Object Dictionary: objects defined by the manufacturer

The first section of this chapter (objects in the interval 2000_h - $2FFF_h$) is dedicated to the internal drive parameters. The second section (objects 3000_h - $3FFF_h$) contains those objects dedicated to sending special low level commands.

4.1 Object 2000_h: Proportional Gain of Current Loop

The object 2000_h is the proportional gain of the current regulation loop.

Description of object	Index	2000_h
	Object name	Proportional Gain of Current Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Proportional gain of the current regulation loop.
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	30 on SM137, 40 on SM140
	Recordable	Yes

4.2 Object 2001_h: Integral Gain of Current Loop

The object 2000_h is the integral gain of the current regulation loop.

Description of object	Index	2001_h
	Object name	Integral Gain of Current Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Integral gain of the current regulation loop.
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	12 on SM137, 10 on SM140
	Recordable	Yes

4.3 Object 2004_h: Maximum Value of Current Loop Output

The object 2004_h is the maximum absolute value of the current regulation loop output.

Description of object	Index	2004_h
	Object name	Maximum value of current loop output
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Maximum value of the current regulation loop output.
	Access	rw (read/write)
	PDO mapping	No
	Unit	Volt x 0,1
	Interval of values	Integer16: 1 - 193 on SM137, : 1 -195 on SM140
	Preset value	193 on SM137, 195 on SM140
	Recordable	Yes

4.4 Object 2005_h: Proportional Gain of Speed Loop

The object 2005_h is the proportional gain of the speed regulation loop.

Description of object	Index	2005_h
	Object name	Proportional Gain of Speed Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Proportional gain of the speed regulation loop.
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	150 on SM137, 600 on SM140
	Recordable	Yes

4.5 Object 2006_h: Integral Gain of Speed Loop

The object 2006_h is the integral gain of the speed regulation loop.

Description of object	Index	2006_h
	Object name	Integral Gain of Speed Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Integral gain of the speed regulation loop.
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	10 on SM137, 50 on SM140
	Recordable	Yes

4.6 Object 2007_h: Percentage Feedforward for Speed Loop

The object 2007_h is the percentage feedforward for the speed regulator.

Description of object	Index	2007_h
	Object name	Percentage Feedforward for Speed Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Percentage feedforward for the speed regulator.
	Access	rw (read/write)
	PDO mapping	No
	Unit	%
	Interval of values	Integer16: 0 - 100
	Preset value	100
	Recordable	Yes

4.7 Object 2009_h: Maximum Value of Speed Loop Output

The object 2009_h is the absolute maximum value of the speed regulation loop output.

Description of object	Index	2009_h
	Object name	Maximum value of speed loop output
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Absolute maximum value of the speed regulation loop output.
	Access	rw (read/write)
	PDO mapping	No
	Unit	Ampere x 0,01
	Interval of values	Integer16: 1 - 900 on SM137, : 1 - 3500 on SM140
	Preset value	500 on SM137, 1800 on SM140
	Recordable	Yes

4.8 Object $200A_h$: Proportional Gain of Position Loop

The object $200A_h$ is the proportional gain of the position regulation loop.

Description of object	Index	$200A_h$
	Object name	Proportional Gain of position Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Proportional gain of the position loop
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	500
	Recordable	Yes

4.9 Object 200B$_h$: Integral Gain of Position Loop

The object 200B$_h$ is the integral gain of the position regulation loop.

Description of object	Index	200B$_h$
	Object name	Integral Gain of position Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00$_h$
	Description	Integral gain of the position loop.
	Access	rw (read/write)
	PDO mapping	No
	Unit	x 0,01
	Interval of values	Integer16: 0 - 32767
	Preset value	0
	Recordable	Yes

4.10 Object 200C$_h$: Percentage Feedforward for Position Loop

The object 200C$_h$ is the percentage feedforward for the position regulator.

Description of object	Index	200C$_h$
	Object name	Percentage Feedforward for position Loop
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00$_h$
	Description	Percentage feedforward for the position regulator.
	Access	rw (read/write)
	PDO mapping	No
	Unit	%
	Interval of values	Integer16: 0 - 100
	Preset value	100
	Recordable	Yes

4.11 Object $200E_h$: Maximum Value of Position Loop Output

The object $200E_h$ is the absolute maximum value of the position regulation loop output.

Description of object	Index	$200E_h$
	Object name	Maximum value of position loop output
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Absolute maximum value of the position loop output.
	Access	rw (read/write)
	PDO mapping	No
	Unit	rpm
	Interval of values	Integer16: 1 - 8000
	Preset value	4500
	Recordable	Yes

4.12 Object $200F_h$: Maximum Speed Following Error

Object $200F_h$ is the maximum speed following error which, when exceeded for a time longer than that foreseen in object 2010_h, will switch the motor to Fault. If the value is 0 the speed following error control is disabled.

Description of object	Index	$200F_h$
	Object name	Maximum Speed Following Error
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Maximum speed following error allowed
	Access	rw (read/write)
	PDO mapping	No
	Unit	rpm
	Interval of values	Integer16: 0 - 8000
	Preset value	0
	Recordable	Yes

4.13 Object 2010_h: Maximum Duration of Speed Following Error

The object 2010_h is the time for which the speed following error must continually exceed the value foreseen in object $200F_h$, in order to switch the motor to Fault.

Description of object	Index	2010_h
	Object name	Maximum Duration of Speed Following Error
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Maximum duration of the speed following error.
	Access	rw (read/write)
	PDO mapping	No
	Unit	Msec
	Interval of values	Integer16: 0 - 10000
	Preset value	0
	Recordable	Yes

4.14 Object 2011_h: Maximum Torque

The object 2011_h is the maximum torque required from the speed regulator. If this is exceeded in a continuous manner for longer than foreseen in object 2012_h, the motor will switch to Fault.

Description of object	Index	2011_h
	Object name	Maximum torque
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Maximum torque
	Access	rw (read/write)
	PDO mapping	No
	Unit	Ampere x 0,01
	Interval of values	Integer16: 0 - 900 on SM137, :35 on SM140
	Preset value	250 on SM137, 900 on SM140
	Recordable	Yes

4.15 Object 2012_h: Timeout for Maximum Torque

The object 2012_h is the time during which the speed regulator torque must continuously exceed the one foreseen in object 2011_h in order to switch the motor to Fault.

Description of object	Index	2012_h
	Object name	Timeout for Maximum Torque
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Timeout for maximum torque
	Access	rw (read/write)
	PDO mapping	No
	Unit	msec
	Interval of values	Integer16: 0 - 10000
	Preset value	1000
	Recordable	Yes

4.16 Object 2013_h: Bit_A

The object 2013_h contains 16 bits, the meanings of which are given in paragraph 4.25.

Description of object	Index	2013_h
	Object name	Bit_A
	Object code	VAR
	Data type	Integer16
	Reference	Paragraph 4.25
Description of elements	Subindex	00_h
	Description	Bit_A
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16
	Preset value	1
	Recordable	Yes

4.17 Object 2014_h: Electric Angle On Zero Index

The object 2014_h is reserved for internal use.

Description of object	Index	2014_h
	Object name	Electric Angle On Zero Index
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Electric Angle On Zero Index
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 799 on SM137, 0 - 1999 on SM140
	Preset value	-
	Recordable	Yes

4.18 Object 2015_h: First Component of Speed Loop Feedforward

The object 2015_h is the first component of the speed regulator feedforward.

Description of object	Index	2015_h
	Object name	First Component of Speed Loop Feedforward
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	First component in the speed regulator feedforward
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 32767
	Preset value	600 on SM137, 300 on SM140
	Recordable	Yes

4.19 Object 2016_h: Second Component of Speed Loop Feedforward

The object 2016_h is the second component of the speed regulator feedforward, the one relating to friction.

Description of object	Index	2016_h
	Object name	Second Component of Speed Loop Feedforward
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Second component of the speed regulator feedforward.
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 32767
	Preset value	1400 on SM137, 600 on SM140
	Recordable	Yes

4.20 Object 2017_h: Third Component of Speed Loop Feedforward

The object 2017_h is the third component of the speed regulator feedforward, the one that is proportional to the speed.

Description of object	Index	2017_h
	Object name	Third Component of Speed Loop Feedforward
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Third component of the speed regulator feedforward.
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 32767
	Preset value	2600 on SM137, 2200 on SM140
	Recordable	Yes

4.21 Object 2018_h: Fourth Component of Speed Loop Feedforward

The object 2018_h is the fourth component of the speed regulator feedforward, the one proportional to acceleration.

Description of object	Index	2018_h
	Object name	Fourth Component of Speed Loop Feedforward
	Object code	VAR
	Data type	Integer16
Description of elements	Subindex	00_h
	Description	Fourth component of the speed regulator feedforward.
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 32767
	Preset value	6 on SM137, 16 on SM140
	Recordable	Yes

4.22 Object 3000_h: Special Commands

With this object it is possible to send special commands to the motor. The following commands are currently available:

Command	Code	Meaning	Param.1	Param.2	Response
CMDNULL	1000_h	Command void: motor is ready to receive commands	-	-	-
CMDGETSMSTAT	$A800_h$	Reads the state of the motor internal automatic device	-	-	Internal motor state
CMDGETSTATAZZ	6400_h	Reads the state of the motor reset	-	-	Motor reset state
CMDSAMPLEVAR	$B400_h$	Samples 2 internal motor variables	First variable to sample	Second variable to sample	-
CMDREADFL	$D400_h$	Reads a flash address	Address to read	-	Value read
CMDWRITEFL	$DC00_h$	Writes a value to a given flash address	Address to write	Value to write	

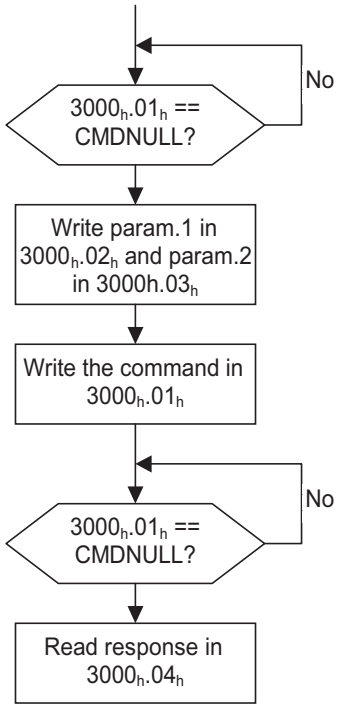
CMDGETDISTMICROZERO (Only on SM140)	$5C00_h$	Reads the distance in counts between the reset switch and the encoder zero notch detected during the last automatic reset operation	-	-	Distance between switch and zero notch

Description of object	Index	3000_h
	Object name	Special commands
	Object code	RECORD
	Data type	-
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 4
	Preset value	4
	Recordable	No

Description of elements	Subindex	01_h
	Description	Command
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16
	Preset value	1000_h
	Recordable	No
	Subindex	02_h
	Description	First command parameter
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer32
	Preset value	0_h
	Recordable	No
	Subindex	03_h
	Description	Second command parameter
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer32
	Preset value	0_h
	Recordable	No
	Subindex	04_h
	Description	Response to the command
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Integer32
	Preset value	0_h
	Recordable	No

Before sending a command by writing the corresponding value in the subindex 01_h it is necessary to read that subindex and check that the value read is CMDNULL, so as to be certain that the motor is ready to receive the command.
Also, before sending the command, it is necessary to write any parameters to subindexes 02_h and 03_h. If the command foresees a response, it is only possible to read the response after the subindex 01_h has once again taken on the value CMDNULL. This procedure is illustrated in Figure 1:

Figure 1



4.23 Object 3010_h: Sampling Variables

With this object it is possible to sample the values of the drive firmware internal variables.

Description of object	Index	3010_h
	Object name	Sampling variables
	Object code	RECORD
	Data type	-
Description of elements	Subindex	00_h
	Description	Maximum subindex supported
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned8: 4
	Preset value	4
	Recordable	No

Description of elements	Subindex	01_h
	Description	First address for the variable
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned16: 60_h $7F_h$, 200_h - $3FF_h$, 800_h - FFF_h
	Preset value	-
	Recordable	No
	Subindex	02_h
	Description	Second address for the variable
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Unsigned16: 60_h $7F_h$, 200_h - $3FF_h$, 800_h - FFF_h
	Preset value	-
	Recordable	No
	Subindex	03_h
	Description	First value for the variable
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16
	Preset value	-
	Recordable	No
	Subindex	04_h
	Description	Second value for the variable
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16
	Preset value	0_h
	Recordable	No

4.24 Object $5f00_h$: Reserved

This object cannot be used.

4.25 Table Bit_A

The following table shows the meanings of the bits in parameter Bit_A.

Bit	Default	Meaning
0	0	If set to 1 this enables control of the software limit switch limits
1	0	Reserved. Leave as 0
2	0	Reserved. Leave as 0
3	0	If set to 1 this reverses the standard direction of rotation of the motor. The positive standard direction of rotation is anticlockwise when viewing the shaft from the flange side
4	0	Reserved. Leave as 0
5	0	Reserved. Leave as 0
6	0	**Only for SM140 and SM137 rev. C** If set to 1 this disables CMDNOREG to exit the state of alarm. In this case the only command to exit the state of alarm is CMDRESET
7	0	Reserved. Leave as 0
8	0	**Only for SM140** If set to 1 this enables the negative out-of-limits cam.
9	0	**Only for SM140** If set to 1 this enables the positive out-of-limits cam.
10	0	**Only for SM140** If set to 1 this forces the motor to switch to alarm state if it encounters one of the two out-of-limits cams. If set to 0 this forces the motor to switch to AXSTOP state, with a suitable deceleration ramp, if it encounters one of the two out-of-limits cams.
11	0	**Version 119 of the firmware and following** Reserved for EnetX: if set to 1 this does not perform the hook-up between bus and regulation
12	0	**Only for SM140 version 119 of the firmware and following.** If set to 0 this forces the motor always to perform controlled braking before leaving regulation. With this bit set to 0, after each alarm condition (with the exception of ALOVERCURR and ALOVERPOWER for which there would be a risk of damaging the drive) or after the command CMDGOEMERG or CMDNOREG the motor checks that its rotation speed is 0. If this is not the case, it performs controlled braking using a deceleration value equal to the value of parameter AMAX. During this phase, if inertia is particularly high, it is necessary to apply a braking resistance to the motor (module P144 supplied by CNI).

5 Objects Dictionary: DSP402

5.1 Object 6040_h: Controlword

The object 6040_h is the Controlword that modifies the state of the drive based on the CiA profile DSP 402. The CANOpen State Machine indicates the operating and error conditions of the drive. A transition in operation, i.e. a change in state between two operating conditions, is regulated by the Controlword. The operating conditions are described by the Statusword (see Object 6041_h).

Description of object	Index	6040_h
	Object name	Controlword
	Object code	VAR
	Data type	Unsigned16
	Reference	CiA DSP-402 V2.0, page 49
Description of elements	Subindex	00_h
	Description	Controlword
	Access	rw (read/write)
	PDO mapping	Yes
	Unit	-
	Interval of values	Unsigned16
	Preset value	0
	Recordable	No
Description of format	Bit 15-13	Not used
	Bit 12	Operation Mode Specific (bit can be customised by manufacturer).
	Bit 11	Warning Acknowledge (bit can be customised by manufacturer).
	Bit 10-9	Reserved
	Bit 8	Halt
	Bit 7	Fault Reset
	Bit 6-4	Operation Mode Specific
	Bit 3	Enable Operation
	Bit 2	Quick Stop
	Bit 1	Enable Voltage
	Bit 0	Switch On

The state is controlled by bits 0 to 3 and by bit 7 of the controlword. The bits marked with an x are irrelevant.

.

Commands	Controlword Bits					Transitions
	Fault Reset Bit 7	Enable Operation Bit 3	Quick Stop Bit 2	Enable Voltage Bit 1	Switch On Bit 0	
Shutdown (stop)	0	x	1	1	0	2,6,8
Switch On	0	x	1	1	1	3
Disable Voltage	0	x	x	0	x	7,9,10,12
Quick Stop	0	x	0	1	x	7,10,11
Disable Operation	0	0	1	1	1	5
Enable Operation	0	1	1	1	1	4,16
Fault Reset	$0 \rightarrow 1$	x	x	x	x	15

For a detailed description of the states and transitions, see the paragraph "DSP402 states" on page 3 of this manual.
Bit 11 "Warning Acknowledge" is used to reset bit 7 of the Statusword: when it is raised this notifies the motor that the presence of a pending message has been detected (signalled by bit 7 high in the Statusword). At this point, bit 7 in the Statusword is lowered.
In Enable Operation state, bits 4 to 6 and bit 8 have different meanings which depend on the type of operating mode (see paragraphs 6.2.1, 6.2.2. and 6.2.3).

5.2 Object 6041_h: Statusword

The object 6041_h is the Statusword which represents the state of the drive according to the CANopen state machine defined in the profile DSP 402. A change in state may be caused either by the Controlword (Object 6040_h) or by internal events in the Smart Motor.

Description of object	Index	6041_h
	Object name	Statusword
	Object code	VAR
	Data type	Unsigned16
	Reference	CiA DSP-402 V2.1, page 50
Description of elements	Subindex	00_h
	Description	Statusword
	Access	ro (read only)
	PDO mapping	Yes
	Unit	-
	Interval of values	Unsigned16
	Preset value	-
	Recordable	No

Description of format	Bit 15-14	Manufacturer specific.
	Bit 13-12	Operating mode specific.
	Bit 11	Internal Limit Active (not implemented)
	Bit 10	Target Reached
	Bit 9	Remote
	Bit 8	Manufacturer specific.
	Bit 7	Warning (message)
	Bit 6	Switch On Disabled
	Bit 5	Quick Stop
	Bit 4	Voltage Enabled
	Bit 3	Fault
	Bit 2	Operation Enabled
	Bit 1	Switched On
	Bit 0	Ready to Switch On

The condition of the machine in CANopen states is represented by bits 0-3, by bit 5 and by bit 6. The following operating conditions are shown by the Statusword (the bits marked with an x are irrelevant):

States	**Statusword bits**					
	Switch on disabled Bit 6	**Quick Stop Bit 5**	**Fault Bit 3**	**Operation Enabled Bit 2**	**Switched on Bit 1**	**Ready to switch on Bit 0**
Not ready to switch on	0	x	0	0	0	0
Switch on disabled	1	x	0	0	0	0
Ready to switch on	0	1	0	0	0	1
Switched on	0	1	0	0	1	1
Operation Enabled	0	1	0	1	1	1
Quick Stop active	0	0	0	1	1	1
Fault reaction active	0	x	1	1	1	1
Fault	0	x	1	0	0	0

Bit 10 and bits 12 and 13 have different meanings according to the type of operating mode. Bit 11, Internal Limit Active piloted by the drive, indicates that an internal limitation is active (for a example software limit in current position). If bit 9 is set, then the parameters can be modified via the CAN network, and the drive will perform the commands contained in the Controlword. If the remote bit is reset, then the drive is in local mode and will not perform the commands. The drive might transmit messages containing valid values according to the configuration of the drive itself. The drive supports the SDO protocol in “local” mode. A drive message is active if bit 7 is set. The presence of the message may be due to an error or to a state that must be mentioned, for example a temperature limit reached or refusal of a command received. The cause of the message must be read by accessing object 1002_h: Manufacturer Status Register. The bit is raised by the device and can be lowered using bit 11 (Warning Acknowledge) in the Controlword. When bit 4 (voltage enable) is set to 1, voltage has been supplied to the drive power supply.

5.2.1 Controlword and Statusword in 'Profile Position Mode'

In "profile position mode" it is possible to perform positioned movements at the speed required. Positions can be absolute or relative with respect to the current position. Furthermore, it is also possible to select whether the motor is to perform the new movement sent immediately, or only at the end of any other movement in progress. If after startup the Smart Motor has never been reset, the positioned movements in Profile Position Mode are not allowed. In this mode the controlword bits involved are 4, 5, 6 and 8:

Name	Bit	Val	Description	Notes
New Set-Point	4	0	Does not use the Target Position.	In Operation Enabled state a positioning operation is made active (value 0→1 or value 1 in input to the Operation Enabled state). The drive acknowledges that the movement command has been received by raising the Setpoint Acknowledged bit in the Statusword (object 6041_h: Statusword bit 12).
		1	Activates the Target Position.	
Change Set Immediately	5	0	Terminates the current positioning and then starts the next one.	If this is set to 1 it allows on-the-spot changing of both the end position and the movement speed.
		1	Interrupts the current positioning and starts the new one.	
abs / rel	6	0	The Target Position is an absolute value.	If it is set to 1, the end position of the movement will be the same as the last Target Position plus the current Target Position. On the other hand, if the Change Set Immediately bit is active, the end position will be the Position Actual Value plus the Target Position.
		1	The Target Position is a relative value.	
Halt	8	0	Performs positioning	A value of 1 allows movement of the drive to be interrupted during the current operation.
		1	Stops the motor using a deceleration ramp	

The Statusword bits involved are 4, 5, 6 and 8:

Name	Bit	Val	Description	Notes
Target Reached	10	0	If Halt =0 the Target Position has not been reached If Halt =1 the motor is performing the deceleration ramp to slow down and stop.	Indicates that the drive has performed a positioning instruction successfully and has reached the Target Position Data; see Object $607A_h$: Target Position, Object 6068_h: Position Window Time and Object 6064_h: Position Actual Value. It will be set even when the drive stops after a stop command.
		1	If Halt=0 the Target Position has been reached If Halt=1 the motor has stopped.	

Name	Bit	Val	Description	Notes
Setpoint Acknowledged	12	0	The trajectory generator has (still) not detected the values for positioning	Notifies the master that the new setpoint has been acquired. A positioning command is started by the "New Set-Point" bit in the controlword. On receiving the command, the drive firmware checks the target position, the operating and regulation parameters and the current state. If the various controls are successful, the firmware sets the "Setpoint acknowledged" bit to 1.
		1	The trajectory generator has received the values for positioning	
Following Error	13	0	No following error	Not yet implemented in the firmware.
		1	Following error	

Example 1: Single movement

A typical operating sequence is illustrated in the table below:

Time		t0	t1	t2	t3	t4	t5	t6
Control word	New Setpoint	0	0	1	1	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	x
	abs/rel	x	x	0	0	x	x	x
Status word	Setpoint Acknowledged	0	0	0	1	1	0	0
	Target Reached	1	1	1	0	0	0	1
Target Position		10	100	100	x	x	x	x
Position Actual Value		10	10	10	11	…	...	100
Velocity Actual Value		0	0	0	50	...	50	0

The motor is stopped in position 10 at start time t0. In t1 the “Target Position” is changed to a value of 100 and immediately after the “New Setpoint” bit is raised. Following this, when the drive has checked all the conditions and is ready to start, the “Setpoint Acknowledged” bit is raised and the “Target Reached” bit is lowered. Note that only when the master lowers the “New Setpoint” bit (time t4) the motor lowers “Setpoint Acknowledged”. When the target position is reached the “Target Reached” bit rises again to communicate the end of the positioning operation.

Example 2: Absolute double movement

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
Control word	New Setpoint	0	0	1	1	0	0	1	1	0	0	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	0	0	x	x	x	x	x
	abs/rel	x	x	0	0	x	x	0	0	x	x	x	x	x
Status word	Setpoint Acknowledged	0	0	0	1	1	0	0	1	1	0	0	0	0
	Target Reached	1	1	1	0	0	0	0	0	0	0	0	0	1

Time	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12
Target Position	10	100	100	x	x	50	50	50	x	x	x	x	x
Position Actual Value	10	10	10	11	…	...	...	...	...	100	99	...	50
Velocity Actual Value	0	0	0	50	...	...	...	...	50	0	-50	...	0

Before the first movement to position 100 terminates, the master requests a new movement at time t6. Simultaneously, the “Change Setpoint Immediately” bit is down, so that the motor waits for the first movement to conclude before running the second. In effect, at time t9 the motor reaches position 100, immediately starts the new movement and does not raise the “Target Reached” bit. The second movement is completed at time t12.

Example 3: Absolute triple movement

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
Controlword	New setpoint	0	0	1	1	0	0	1	1	0	0	1
	Change Setpoint Immediately	x	x	x	x	x	x	0	0	x	x	0
	abs/rel	x	x	0	0	x	x	0	0	x	x	0
Statusword	Setpoint Acknowledged	0	0	0	1	1	0	0	1	1	0	0
	Target Reached	1	1	1	0	0	0	0	0	0	0	0
Target Position		10	100	100	x	x	50	50	50	x	x	75
Position Actual Value		10	10	10	11	…	...	...	...	...	...	...
Velocity Actual Value		0	0	0	50	...	...	...	...	...	...	50

Time		t11	t12	t13	t14	t15	t16	t17	t18	t19
Controlword	New Setpoint	1	1	1	0	0	0	0	0	0
	Change Setpoint Immediately	0	0	0	x	x	x	x	x	x
	abs/rel	0	0	0	x	x	x	x	x	x
Statusword	Setpoint Acknowledged	0	0	1	1	0	0	0	0	0
	Target Reached	0	0	0	0	0	0	0	0	0
Target Position		75	x	x	x	x	x	x	x	1
Position Actual Value		100	99	...	...	...	75	74	...	50
Velocity Actual Value		0	-50	...	...	-50	0	-50	...	0

If, before completing the first movement (final position 100), the master requests a second one at time t6 (final position 75) and then a third one at time t10 (final position 50), the motor is unable to receive the latter positioning, as it only has a single element queue. It will therefore not raise the “Setpoint Acknowledge” bit until immediately after it has terminated the first movement (time t13).

Example 4: Absolute double movement with “Change Setpoint Immediately” active

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
Control word	New Setpoint	0	0	1	1	0	0	1	1	0	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	1	1	x	x	x	x
	abs/rel	x	x	0	0	x	x	0	0	x	x	x	x
Status word	Setpoint Acknowledged	0	0	0	1	1	0	0	1	1	0	0	0
	Target Reached	1	1	1	0	0	0	0	0	0	0	0	1
Target Position		10	100	100	x	x	50	50	50	x	x	x	x
Position Actual Value		10	10	10	12	…	60	64	66	66	64	...	50
Velocity Actual Value		0	0	0	50	...	100	100	50	0	-50	...	0

Before the first movement to position 100 ends, the master requests a new movement at time t6 with the “Change Setpoint Immediately” bit set to 1. In this case the motor has reached position 64 and has to slow down and reverse to reach position 50. Deceleration starts at time t7 (you can see the speed dropping and changing from positive to negative).

Example 5: Relative single movement

Time		t0	t1	t2	t3	t4	t5	t6
Controlword	New Setpoint	0	0	1	1	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	x
	abs/rel	x	x	1	1	x	x	x
Statusword	Setpoint Acknowledged	0	0	0	1	1	0	0
	Target Reached	1	1	1	0	0	0	1
Target Position		10	100	100	x	x	x	x
Position Actual Value		10	10	10	11	...	...	110
Velocity Actual Value		0	0	0	50	...	50	0

In this case the movement is incremental and the final position is therefore equal to:

last Target Position + Target Position = 10 + 100 = 110

Example 6: Relative double movement

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
Control word	New Setpoint	0	0	1	1	0	0	1	1	0	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	0	0	x	x	x	x
	abs/rel	x	x	1	1	x	x	1	1	x	x	x	x
Status word	Setpoint Acknowledged	0	0	0	1	1	0	0	1	1	0	0	0
	Target Reached	1	1	1	0	0	0	0	0	0	0	0	1
Target Position		10	100	100	x	x	x	120	120	x	x	x	x

Time	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
Position Actual Value	10	10	10	11	…	...	...	...	...	110	...	230
Velocity Actual Value	0	0	0	50	...	...	...	...	50	0	50	0

In this case the movement is incremental and the final position of the second movement is equal to:

last Target Position + Target Position = 100 + 120 = 230

Example 7: Relative double movement with “Change Setpoint Immediately” active

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11
Control word	New Setpoint	0	0	1	1	0	0	1	1	0	0	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	1	1	x	x	x	x
	abs/rel	x	x	1	1	x	x	1	1	x	x	x	x
Status word	Setpoint Acknowledged	0	0	0	1	1	0	0	1	1	0	0	1
	Target Reached	1	1	1	0	0	0	0	0	0	0	0	1
Target Position		10	100	100	x	x	x	120	120	x	x	x	x
Position Actual Value		10	10	10	11	…	...	70	72	...	...	...	192
Velocity Actual Value		0	0	0	50	...	...	100	100	100	...	50	0

The second movement is incremental and with immediate change of the Setpoint. In this specific case the final position is equal to:
Position Actual Value + Target Position = 72 + 120 = 192

Example 8: Single movement with stop

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8
Controlword	New Setpoint	0	0	1	1	0	0	x	0	0
	Change Setpoint Immediately	x	x	x	x	x	x	x	x	x
	abs/rel	x	x	0	0	x	x	x	x	x
Status word	Setpoint Acknowledged	0	0	0	0	0	0	1	0	0
	target reached	0	0	0	1	1	0	0	0	0
Target Position		1	1	1	0	0	0	0	0	1
Position Actual Value		10	100	100	x	x	x	x	x	x
Velocity Actual Value		10	10	10	11	…	40	42	43	43
Control word	New Setpoint	0	0	0	50	...	100	100	50	0

At time t6 the master raises the Halt bit to stop the motor. As soon as possible a braking operation is carried out, bringing the motor to a position that is different from the set Target Position. At the

end of the braking operation the motor raises the “Target Reached” bit. If a double “Setpoint” were to have been sent, the Halt would also have aborted this second movement. If the Halt bit is returned to 0 during the braking ramp, it is possible to send the motor another movement, which will be carried out as soon as braking has terminated.

5.2.2 Controlword and Statusword in 'Profile Velocity Mode'

In this mode it is possible to carry out movements at the requested velocity without a target position: unless the motor is explicitly stopped, it will continue to turn indefinitely. In Profile Velocity Mode The only Controlword bit involved is number 8:

Name	Bit	Val	Description	Notes
Halt	8	0	Performs the movement	In Operation Enabled state the Halt bit set to 1 will result in stoppage of the motor. The drive communicates the fact that it is stopped using the bit Speed $\neq 0$ in the Statusword.
		1	Stops the axis	

The Statusword bits involved are numbers 4, 5, 6 and 8:

Name	Bit	Val	Description	Notes
Target Reached	10	0	If Halt =0 Target Velocity not yet reached If Halt =1 the motor is slowing down before stopping.	The Target Reached bit indicates that the guide drive has reached the given Target Velocity; see object $60FF_h$ and object $606C_h$. The bit is also set when the motor stops after a Halt command.
		1	If Halt=0 the Target Velocity has been reached Halt=1 the motor is stopped.	
Speed $\neq 0$	12	0	Speed $\neq 0$	The bit Speed = 0 is reset (value $1 \rightarrow 0$) if the drive rpm exceed a minimum amount.
		1	Speed = 0	
Max Slippage Error	13	0	Maximum slippage not reached	Not yet implemented in the firmware
		1	Maximum slippage reached	

Example 1: Movement with variation of velocity and halt

A typical operating sequence is illustrated in the following table. Reading the table from left to right it is possible to note the changes in the values of objects as time passes:

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13
Controlword	Halt	1	0	0	0	0	0	0	0	0	1	1	1	1	1
Statusword	Target Reached	1	1	0	0	1	1	0	0	1	1	0	0	0	1
	Speed $\neq 0$	1	1	1	0	0	0	0	0	0	0	0	0	1	1
Target Velocity		x	100	100	100	100	200	200	200	200	x	x	x	x	x

Time	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13
Velocity Actual Value	0	0	1	...	100	100	101	...	200	200	199	...	1	0

At the time t1 the Halt bit is lowered by the master and the motor starts to move, lowering the “Target Reached” bit (an acceleration ramp is in progress and therefore the Target Velocity has not yet been reached). At time t4 the motor has terminated its ramp and the “Target Reached” bit is raised accordingly.
At time t5 the Target Velocity is changed by the master, and the motor therefore starts another ramp to reach the new speed.
At time t9 the master raises the Halt bit, requesting the motor to stop. A deceleration ramp is started which will cause the motor to stop at time t13.

5.2.3 Controlword and Statusword in 'Homing Mode'

In this mode it is possible to reset the smart motor.
In homing mode the Controlword bits involved are numbers 4, 8 and 12 :

Name	Bit	Val	Description	Notes
Homing Operation Start	4	0	Homing Mode not active	In Operation Enabled state, the Homing Operation Start bit commands the drive to start the position reset procedure. The Smart Motor communicates the end of the reset operation using the Homing Attained bit in the Statusword.
		0→1	Starts Homing Mode	
		1	Homing Mode active	
		1→0	Stops Homing Mode	
Halt	8	0	Performs the instruction in bit 4	A value of 1 in the Halt bit allows movement of the drive to be stopped during reset. After the bit has been reset (value 1→0) the reset starts again.
		1	Stops the axis	
Latch Zero Index	12	0	Waits before detecting the encoder zero index.	
		1	Enables detection of the zero index for the reset operation.	

The Statusword bits involved are numbers 10 and 12:

Name	Bit	Val	Description	Notes
Target Reached	10	0	If Halt =0 the Home Position has not been reached. If Halt =1 it is slowing down before stopping.	
		1	If Halt =0 the Home Position has been reached. If Halt =1 the motor is stopped.	

Name	Bit	Val	Description	Notes
Homing Attained	12	0	Reset not completed	The Homing Attained bit indicates that the drive has stopped after completing the reset. See (Homing Method), (Homing Speeds) and (Homing Offset).
		1	Reset completed successfully	

There are two possible types of reset on the SM137 (see object 6098_h: Homing method):

- Manual reset (Homing Method = 35): the current position is assigned the value preset in object $607C_h$: Homing Offset. It is obvious that this type of reset is difficult to relate to a specific mechanical position of the motor, unless you perform a "stop" reset: i.e. the motor is brought into contact with an obstacle and then this position is assigned the required value.
- Automatic reset (Homing Method = -1 or -2): the motor position is assigned the Homing Offset value at the moment in which the encoder zero index is read. The precision of this type of reset is without doubt superior to the one described above.
  This type of reset involves three phases:
  - the motor starts to move at the speed assigned in the object 6099_h: Homing Speed-Value. The direction of rotation is determined by the type of reset: if Homing Method = -1 then the motor moves in the negative direction, if Homing Method = -2 then the motor moves in the positive direction.
  - at the moment in which the Latch Zero Index bit changes to 1 the motor looks for the encoder zero index and on encountering the first index assigns the value written in object $607C_h$: Homing Offset to its position.
  - the motor starts to slow down until stopping. The reset operation is complete

Please note: raising the Halt bit during the reset operation only results in a pause in the reset itself, which will start up again when the bit is lowered. On the other hand, if the Homing Start bit is lowered before the reset operation has been completed, the reset will be aborted.

Example 1: Manual reset (Homing Method = 35).

The following table shows the progress of certain objects in the time during a manual reset (Homing Offset = 35):

Time		t0	t1	t2	t3
Controlword	Homing Start	0	1	x	x
	Halt	0	0	0	0
	Latch Zero Index	x	x	x	x
Statusword	Target Reached	1	1	0	1
	Homing Attained	x	x	0	1
Homing Offset		80	80	80	80
Position Actual Value		10	10	10	80
Velocity Actual Value		0	0	0	0

At time t1 the start of the reset operation is commanded. Within a few moments, at time t2, the Target Reached and Homing Attained bits switch to 0 (the reset has started but has not yet been completed) and immediately after this, at time t3, both bits return to 1.

Example 2: automatic reset (Homing Method = -1 or -2).

Time		t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
Control word	Homing Start	0	1	1	1	1	1	1	1	x	x	x
	Halt	x	0	0	0	0	0	0	0	0	0	0
	Latch Zero Index	x	0	0	0	1	x	x	x	x	x	x
Status word	Target Reached	1	1	0	0	0	0	0	0	0	0	1
	Homing Attained	x	x	0	0	0	0	0	0	0	0	1
Homing Offset		80	80	80	80	80	80	80	80	80	80	80
Position Actual Value		10	10	11	...	101	105	...	80	83	...	86
Velocity Actual Value		0	0	5	...	20	20	20	20	15	...	0

5.3 Object 6060_h: Modes of operation

The object 6060_h regulates the control guide operating mode. The Smart Motor uses the following operating modes:

- Profile Position Mode (Reference CiA DSP-402)
- Profile Velocity Mode (Reference CiA DSP-402)
- Homing Mode (Reference CiA DSP-402)

Description of object	Index	6060_h
	Object name	Modes of operation
	Object code	VAR
	Data type	Integer8
	Reference	CiA DSP-402 V2.0, page 68
Description of elements	Subindex	00_h
	Description	Modes of operation
	Access	wo (write only)
	PDO mapping	Yes
	Unit	-
	Interval of values	Integer8: 0, 1, 3, 6
	Preset value	0
	Recordable	Yes
Description of data	0	OFF (no mode selected)
	1	Profile Position Mode
	2	Velocity mode (not supported)
	3	Profile velocity mode
	4	Torque profile mode (not supported)
	5	reserved
	6	Homing mode
	7	Interpolated Position mode (not supported)
	8...127	reserved
	-1...-128	Manufacturer specific (not supported)

Profile Position Mode

In Profile Position Mode (value 1) it is possible to carry out positioned movements at variable speeds.
The following parameters can be modified to change the characteristics of the movement and monitor its progress:

- Object 6040_h: Controlword
- Object 6041_h: Statusword
- Object 6065_h: Following Error Window
- Object 6066_h: Following Error Timeout
- Object 6064_h: Position Actual Value
- Object $607A_h$: Target Position
- Object 6081_h: Profile Velocity
- Object 6083_h: Profile Acceleration

See paragraph 5.2.2 “Controlword and Statusword in 'Profile Position Mode'”.
If the Smart Motor has never been reset since starting, positioned movements in Profile Position Mode are not allowed.

Profile Velocity Mode

In Profile Velocity Mode (value 3), it is possible to carry out movements at variable velocity without a final target position.
The following operating parameters can be modified or tested during Profile Velocity Mode:

- Object 6040_h: Controlword
- Object 6041_h: Statusword
- Object $606C_h$: Velocity Actual Value
- Object 6083_h: Profile Acceleration
- Object $60FF_h$: Target Velocity

See paragraph 5.2.2 “Controlword and Statusword in 'Profile Velocity Mode'”.

Homing Mode

Homing Mode (value 6) allows the position of the motor to be initialised. Reset takes place both via a movement to detect the encoder zero index and by assigning the motor a position.
The following operating parameters can be set and/or monitored during Homing Mode:

- Object 6040_h: Controlword
- Object 6041_h: Statusword
- Object $607C_h$: Homing Offset
- Object 6083_h: Profile Acceleration
- Object 6098_h: Homing Method
- Object 6099_h: Homing Speed Value

See paragraph 5.2.3 “Controlword and Statusword in 'Homing Mode'”.
If the Smart Motor has never been reset since starting, positioned movements in Profile Position Mode are not allowed.

5.4 Object 6061_h: Modes of operation display

The object 6061_h shows the current operating method. The meaning of the return value corresponds to object 6060_h: Modes of operation.

Description of object	Index	6061_h
	Object name	Modes of operation display
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 56
Description of elements	Subindex	00_h
	Description	Active operating mode
	Access	ro (read only)
	PDO mapping	Yes
	Unit	-
	Interval of values	Integer8
	Preset value	No
	Recordable	No
Description of data	0	OFF (no mode selected)
	1	Profile Position Mode
	2	Velocity mode (not supported)
	3	Profile velocity mode
	4	Torque profile mode (not supported)
	5	reserved
	6	Homing mode
	7	Interpolated Position mode (not supported)
	8...127	reserved
	-1...-128	Manufacturer specific (not supported)

5.5 Object 6064_h: Position Actual Value

The object 6064_h shows the current position of the Smart Motor in the logic memory position. The unit of measurement for this data item is the encoder increment.

Description of object	Index	6064_h
	Object name	Position Actual Value
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 99
Description of elements	Subindex	00_h
	Description	Current position value
	Access	ro (read only)
	PDO mapping	Yes
	Unit	increment
	Interval of values	Integer32
	Preset value	No
	Recordable	No

5.6 Object 6065_h: Following Error Window

The object 6065_h regulates a symmetrical position tolerance for control of the following error. A value of 2^{32} - 1 for the following error window disables the following error control. The following error window works together with object 6066_h: Following Error Timeout.

Description of object	Index	6065_h
	Object name	Following Error Window
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 100
Description of elements	Subindex	00_h
	Description	Following error window
	Access	rw (read/write)
	PDO mapping	No
	Unit	encoder counts
	Interval of values	Unsigned32: 1 - $7FFFFFFF_h$, $FFFFFFFF_h$
	Preset value	4294967295
	Recordable	Yes

5.7 Object 6066_h: Following Error Timeout

The object 6066_h sets a time window to check the following error. The following error timeout works together with object 6065_h: Following Error Window.

Description of object	Index	6066_h
	Object name	Following Error Timeout
	Object code	VAR
	Data type	Unsigned16
	Reference	CiA DSP-402 V2.0, page 100
Description of elements	Subindex	00_h
	Description	Following error timeout
	Access	rw (read/write)
	PDO mapping	No
	Unit	milliseconds
	Interval of values	Unsigned16: 0 - 1000
	Preset value	0
	Recordable	Yes

If the motor commits a following error higher than the Following Error Window for a continuous period of time exceeding the Following Error Timeout, the motor will switch to a state of Fault (see "DSP402 states") and the 16 lowered bits of object 1002_h: Manufacturer Status Register will contain the value 6 corresponding to ALMAXERRORP.

5.8 Object 6067_h: Position Window

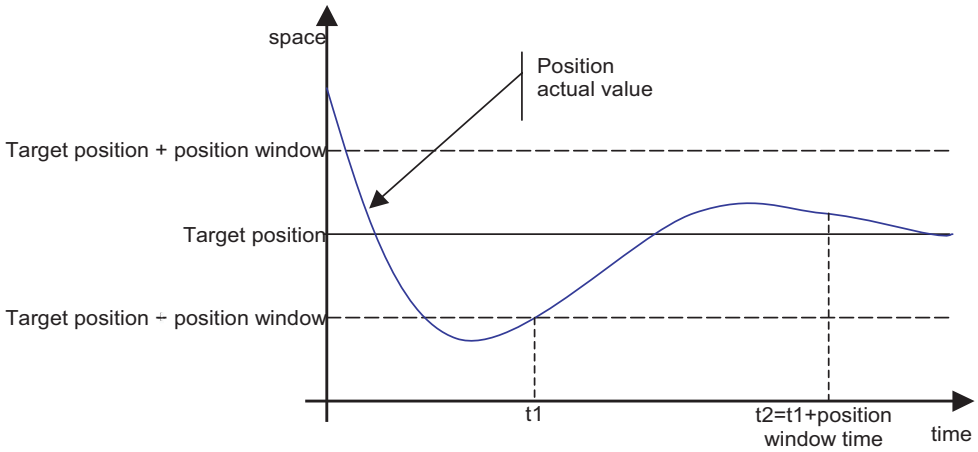
The Position Window defines a symmetrical interval of acceptable positions around the Target Position. If the current value of the position encoder is within the Position Window, the Target Position is considered to have been reached. If the value of the Position Window is 2^{32} - 1, the position of the control window is disabled. The Position Window works in combination with object 6068_h: Position Time Window.

Description of object	Index	6067_h
	Object name	Position Window
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 100
Description of elements	Subindex	00_h
	Description	Position window
	Access	rw (read/write)
	PDO mapping	No
	Unit	encoder counts
	Interval of values	Unsigned32: 1 - $7FFFFFFF_h$, $FFFFFFFF_h$
	Preset value	4294967295
	Recordable	Yes

Bit 10 of the object 6041_h: Statusword (Target Reached) is not raised until, at the end of a positioning operation, the Position Actual Value falls continuously within the symmetrical area

around the Target Position with an extent equal to the Position Window for the length of time defined in the Position Window Time.
The figure below illustrates its operation:

Figure 1



At time t1 the actual position of the axis falls within the interval around the Target Position and until time t2, which is equal to t1 + Position Window Time, it does not leave this interval. At time t2 the Smart motor raises bit 10 of the Statusword (Target Reached) so as to declare that the motor has reached its position.

5.9 Object 6068_h: Position Window Time

When the actual position is within the Position Window for a time exceeding the Position Window Time, bit 10 (Target Reached) of the Statusword is set to 1.

Description of object	Index	6068_h
	Object name	Position Window Time
	Object code	VAR
	Data type	Unsigned16
	Reference	CiA DSP-402 V2.0, page 101
Description of elements	Subindex	00_h
	Description	Position window time
	Access	rw (read/write)
	PDO mapping	No
	Unit	milliseconds
	Interval of values	Unsigned16: 0 - 10000
	Preset value	0
	Recordable	Yes

5.10 Object $606B_h$: Velocity Demand Value

The object $606B_h$ shows the velocity required of the speed regulator by the position regulator. This information may be used for optimisation of the regulation parameters.

Description of object	Index	$606B_h$
	Object name	Velocity Demand Value
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 126
Description of elements	Subindex	00_h
	Description	Requested velocity value
	Access	ro (read only)
	PDO mapping	Yes
	Unit	rpm
	Interval of values	Integer32
	Preset value	No
	Recordable	No

5.11 Object $606C_h$: Velocity Actual Value

The object $606C_h$ shows the current velocity of the motor.

Description of object	Index	$606C_h$
	Object name	Velocity Actual Value
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 126
Description of elements	Subindex	00_h
	Description	Velocity Actual Value
	Access	ro (read only)
	PDO mapping	Yes
	Unit	rpm
	Interval of values	Integer32
	Preset value	No
	Recordable	No

5.12 Object $607A_h$: Target Position

The Target Position is the position that the motor has to reach in Profile Position Mode. The Target Position will be interpreted as an absolute or relative value according to the value of the ‘abs/rel' bit in the Controlword.

Description of object	Index	$607A_h$
	Object name	Target Position
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 78
Description of elements	Subindex	00_h
	Description	Target position
	Access	rw (read/write)
	PDO mapping	Yes
	Unit	encoder counts
	Interval of values	Integer32
	Preset value	0
	Recordable	Yes

5.13 Object $607C_h$: Homing Offset

This object contains the position assigned to the current position during a manual reset (Homing Method 35) or the value assigned to the encoder zero index during an automatic reset operation (Homing Method -1 or -2). See paragraph 5.2.3 “Controlword and Statusword in Homing Mode”.

Description of object	Index	$607C_h$
	Object name	Homing Offset
	Object code	VAR
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 88
Description of elements	Subindex	00_h
	Description	Homing offset
	Access	rw (read/write)
	PDO mapping	No
	Unit	encoder counts
	Interval of values	Integer32
	Preset value	0
	Recordable	Yes

5.14 Object 607D$_h$: Software Position Limit

Software Position Limit contains the sub-parameters Min-Position Limit and Max-Position Limit. These parameters define the absolute position limits for the Position Demand Value and for the Position Actual Value. Each new Target Position must be restricted within these limits.
If the Smart Motor has never been reset since startup, control of the software limits is not enabled. Furthermore, in Homing Mode also the software limits control is not enabled.

Description of object	Index	607D$_h$
	Object name	Software Position Limit
	Object code	ARRAY
	Data type	Integer32
	Reference	CiA DSP-402 V2.0, page 80
Description of elements	Subindex	00$_h$
	Description	Number of elements
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Integer32
	Preset value	2
	Recordable	No
Description of elements	Subindex	01$_h$
	Description	Minimum position limit
	Access	rw (read/write)
	PDO mapping	No
	Unit	encoder counts
	Interval of values	Integer32
	Preset value	No
	Recordable	Yes
Description of elements	Subindex	00$_h$
	Description	Maximum position limit
	Access	rw (read/write)
	PDO mapping	No
	Unit	encoder counts
	Interval of values	Integer32
	Preset value	No
	Recordable	Yes

5.15 Object 6081_h: Profile Velocity

The Profile Velocity is the speed normally obtained at the end of the acceleration ramp during a movement in Profile Position Mode, and it is valid for both directions of movement.

Description of object	Index	6081_h
	Object name	Profile Velocity
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 81
Description of elements	Subindex	00_h
	Description	Velocity profile
	Access	rw (read/write)
	PDO mapping	No
	Unit	rpm
	Interval of values	Unsigned32: 0...8000
	Preset value	0
	Recordable	Yes

5.16 Object 6083_h: Profile Acceleration

The object 6083_h sets acceleration of the drive movements in Profile Position Mode, Profile Velocity Mode and Homing Mode.

Description of object	Index	6083_h
	Object name	Profile Acceleration
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 82
Description of elements	Subindex	00_h
	Description	Acceleration profile
	Access	rw (read/write)
	PDO mapping	No
	Unit	10000 counts/sec^2
	Interval of values	Unsigned32: 1...32767
	Preset value	32
	Recordable	Yes

5.17 Object 6085_h: Quick Stop Deceleration

The Quick Stop Deceleration is the deceleration used to stop the motor when the 'Quick Stop' command is given.

Description of object	Index	6083_h
	Object name	Quick Stop Deceleration
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 83
Description of elements	Subindex	00_h
	Description	Quick stop deceleration
	Access	rw (read/write)
	PDO mapping	No
	Unit	10000 counts/sec^2
	Interval of values	Unsigned32: 1...32767
	Preset value	48
	Recordable	Yes

5.18 Object 6098_h: Homing Method

The object Homing Method determines the type of reset that will be used. There are three reset methods implemented, and they are described in paragraph 5.2.3 "Controlword and Statusword in Homing Mode".
Until the Smart Motor is reset it is not possible to perform positioned movements in Profile Position Mode and also the software limits control is not active.

Description of object	Index	6083_h
	Object name	Homing Method
	Object code	VAR
	Data type	Integer8
	Reference	CiA DSP-402 V2.0, page 89
Description of elements	Subindex	00_h
	Description	Homing method
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer8: -1, -2, 35
	Preset value	0
	Recordable	No

5.19 Object 6099_h: Homing Speed Value

Defines the speeds used during the reset operation.

Description of object	Index	6099_h
	Object name	Homing Speeds
	Object code	ARRAY
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 90
Description of elements	Subindex	00_h
	Description	Number of elements
	Access	ro (read only)
	PDO mapping	No
	Unit	-
	Interval of values	Integer32: 1
	Preset value	1
	Recordable	No
	Subindex	01_h
	Description	Speed during search for reset micro.
	Access	rw (read/write)
	PDO mapping	No
	Unit	rpm
	Interval of values	Unsigned32: 1...8000
	Preset value	500
	Recordable	Yes

5.20 Object $60FF_h$: Target Velocity

The object $60FF_h$ sets the Target Velocity of the movement in Profile Velocity Mode.

Description of object	Index	$60FF_h$
	Object name	Target Velocity
	Object code	VAR
	Data type	Unsigned32
	Reference	CiA DSP-402 V2.0, page 128
Description of elements	Subindex	00_h
	Description	Target velocity
	Access	rw (read/write)
	PDO mapping	Yes
	Unit	rpm
	Interval of values	Integer32: -8000...8000
	Preset value	0
	Recordable	Yes

6 Objects that are not implemented

By changing the value of objects in the DSP 402 it is possible to modify the reaction of the motor in certain special conditions. Some of these objects have not been implemented, so the corresponding reaction of the motor has been preset by CNI and in the current version of the firmware it cannot be modified. The following shows which these preset reactions are.

6.1 Object 6007_h: Abort Connection Option Code

The object 6007_h regulates the reaction of the drive to a loss of connection to the NMT-Master (Life Guarding Event).

Description of object	Index	6007_h
	Object name	Abort Connection Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 34
Description of elements	Subindex	00_h
	Description	Connection broken code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16: 0 - 3
	Preset value	0
	Recordable	Yes
Description of data	0	no action (not used)
	1	malfunction
	2	command controlling the ‘disabled voltage’ device (not used)
	3	command controlling the ‘quick stop’ device (not used)
	4...32767	reserved (not used)
	-32768...-1	production specifications (not used)

The motor reacts as it would if this object were set to a value of 1:
The Life-Guarding Event is considered a serious error, so the state of the drive is switched to Fault. Movement of the motor is stopped and the power electronics are disabled (see object $605E_h$: Fault Reaction Option Code).

6.2 Object $605A_h$: Quick Stop Option Code

The parameter Quick Stop Option Code determines what action is taken when the Quick Stop function is performed.

Description of object	Index	$605A_h$
	Object name	Quick Stop Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 53
Description of elements	Subindex	00_h
	Description	Quick stop code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16:
	Preset value	2
	Recordable	Yes
Description of data	0	Disables the driving function (not supported)
	1	Slows down on the deceleration ramp (not supported)
	2	Slows down on the quick stop ramp (not supported)
	3	Lowers the current limit (not supported)
	4	Lowers the voltage limit (not supported)
	5	Slows down on the deceleration ramp and remains in Quick-Stop (not supported)
	6	Slows down on the quick stop ramp and remains in Quick-Stop
	7	Lowers the current limit and remains in Quick-Stop (not supported)
	8	Lowers the voltage limit and remains in Quick-Stop (not supported)
	9...32767	Reserved (not supported)
	-1...-32768	Production specifications (not supported)

The motor reacts as it would if this object were set to a value of 6.

6.3 Object $605C_h$: Disable Operation Option Code

The object $605C_h$ regulates the reaction of the drive to a Disable Operation command.

Description of object	Index	$605C_h$
	Object name	Disable Operation Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 53
Description of elements	Subindex	00_h
	Description	Disable operation option code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16:
	Preset value	0
	Recordable	Yes
Description of data	0	Disables the driving function (not supported)
	1	Slows down on the deceleration ramp and disables the driving function
	2...32767	Reserved (not supported)
	-1...-32768	Production specifications (not supported)

The motor reacts as it would if this object were set to a value of 1: the motor brakes using the normal deceleration ramp (object 6083_h: Profile Acceleration). After this the power electronics are disabled.

6.4 Object $605B_h$: Shutdown Option Code

The parameter Shutdown Option Code determines which action should be taken with a Shutdown command.

Description of object	Index	$605B_h$
	Object name	Shutdown Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 52
Description of elements	Subindex	00_h
	Description	Shutdown code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16:
	Preset value	0
	Recordable	Yes
Description of data	0	Disables adjustment, the motor is free to turn
	1	Slows down on the deceleration ramp (not supported)
	2	Slows down on the quick stop ramp (not supported)
	3	Lowers the current limit (not supported)
	4	Lowers the voltage limit (not supported)
	5...32767	Reserved (not supported)
	-1...-32768	Production specifications (not supported)

The motor reacts as it would if this object were set to a value of 0:
Setting the parameter to a value of 0 disconnects the power electronics with a Shutdown order, i.e. the motor is still free to turn.

6.5 Object $605D_h$: Halt Option Code

The parameter Halt Option Code determines what action will be taken if bit 8 (halt) of the Controlword is active.

Description of object	Index	$605D_h$
	Object name	Halt Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 54
Description of elements	Subindex	00_h
	Description	Halt code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16:
	Preset value	1
	Recordable	Yes
Description of data	0	Disables driving, the motor is free to turn (not supported)
	1	Slows down on the deceleration ramp
	2	Slows down on the quick stop ramp (not supported)
	3	Lowers the current limit (not supported)
	4	Lowers the voltage limit (not supported)
	5...32767	Reserved (not supported)
	-1...-32768	Production specifications (not supported)

The motor reacts as it would if this object were set to a value of 1: the motor brakes using the normal deceleration ramp (object 6083_h: Profile Acceleration).

6.6 Object $605E_h$: Fault Reaction Option Code

The parameter Fault Reaction Option Code determines what action will be taken if an error occurs on the drive.

Description of object	Index	$605E_h$
	Object name	Fault Reaction Option Code
	Object code	VAR
	Data type	Integer16
	Reference	CiA DSP-402 V2.0, page 54
Description of elements	Subindex	00_h
	Description	Fault reaction option code
	Access	rw (read/write)
	PDO mapping	No
	Unit	-
	Interval of values	Integer16:
	Preset value	2
	Recordable	Yes
Description of data	0	Disables adjustment, the motor is free to turn
	1	Slows down on the deceleration ramp (not supported)
	2	Slows down on the quick stop ramp (not supported)
	3	Lowers the current limit (not supported)
	4	Lowers the voltage limit (not supported)
	5...32767	Reserved (not supported)
	-1...-32768	Production specifications (not supported)

The motor reacts as it would if this object were set to a value of 0:
the power electronics are disabled and the motor is brought to a stop at 0 rpm, i.e. the motor is still free to turn.

7 Emergency Messages

The emergency messages are emitted when faults occur on the drive, and they are assigned the highest possible priority to ensure that they access the bus without any wait. An Emergency Object is only transmitted once for each fault that occurs. No further emergency object will be transmitted until another fault occurs on a device. The emergency messages contain an error field with preset error codes and additional information. The emergency telegram consists of 8 bytes with the data shown in the following figure:

Byte	Content
0	Emergency Error Code
1	
2	Error Register (object 1001_h)
3	Manufacturer specific fault field
4	
5	
6	
7	

(see DS 301 Version 4.02 page 60 and DSP 402 Version 2.0 page 19)

The Emergency Error Codes for the Smart Motor are described in the following table:

Error code (hex)	Meaning	Defined by	Notes
0000	No error	DS 301	If the startup phase has been successful the Smart Motor sends this message when the reset takes place.
2130	Shortcircuit	DSP 402	Too much current is circulating in the power circuit. Probable shortcircuit. It is necessary to repair the motor.
2220	Continuous internal overcurrent	DSP 402	The current required for the motor has exceeded, for a period longer than the time foreseen in object 2012_h, the torque set in object 2011_h. It is necessary to decrease the load and/or the motor acceleration or reset the parameters.
3110	Main over-voltage	DSP 402	The power system supply voltage is too high, and there is a risk of damaging the motor. If the problem is encountered during the motor deceleration phase, an electric brake should be installed.
3120	Main under-voltage	DSP 402	The power system supply voltage has dropped to below 16 Volt while the motor was under regulation. The motor must be taken out of regulation before the power system supply is disconnected, or a power feeder of suitable capacity must be installed

Error code (hex)	Meaning	Defined by	Notes
4310	Drive temperature too high	DSP 402	The drive has overheated. The application is two much for the motor: reduce the inertia and/or the accelerations and/or the percentage use of the motor.
6310	Software device: parameters lost	DSP 402	The parameters saved in the motor flash memory have been lost. The motor will start up with its default parameters. It will be necessary to set the motor parameters again.
8100	Control communication error	DS 301	A bus off or an overrun has occurred during transmission of SDO messages. Check the motor wiring and/or check for excessive traffic on the CAN network.
8110	Communication error: CAN overrun (objects lost)	DS 301	An overrun has occurred while receiving NMT and/ or SDO packages.
8130	Communication error: life guard error	DS 301	The time passed during node guarding between two successive packages is higher than the one entered in object $100C_h$ multiplied by the one in $100D_h$.
8140	Communication recovered after bus-off	DS 301	The bus-off condition has ended and the Smart Motor attempts to restore the communication.
8210	Protocol error: the PDO has not been processed due to an error in its length	DS 301	The motor has received a PDO package with an incorrect length.
8611	Controller positioning: following error	DSP 402	A position following error higher than the value in 6065_h has been detected continuously for longer than foreseen in 6066_h. The motor has encountered an obstacle or the parameters need to be calibrated.
FF00	Internal error: trajectory generator	Supplier	An error has occurred in the trajectory generator: contact the supplier.
FF01	Speed controller: following error	Supplier	A velocity following error higher than the value in $200F_h$ has been detected continuously for longer than foreseen in 2010_h. The motor has encountered an obstacle or the parameters need to be calibrated.
FF02	Internal error: unexpected interrupt	Supplier	An internal error has occurred in the firmware: contact the supplier.

8 SDO Interrupt Codes

The protocol Abort SDO Transfer interrupts an SDO transfer due to an error. The following table lists all the SDO interrupt codes that may occur during data exchanged via the SDO service (see DS301 version 4.02 page 48).

Interrupt Code (hex)	Description
05030000	Toggle bit not alternate.
05040001	Client/server command specified is invalid or unknown.
06010000	Access to an object not supported.
06010001	Attempting to read a write-only object.
06010002	Attempting to write a read-only object.
06020000	The object does not exist in the objects dictionary.
06040041	The object cannot be mapped in the PDO.
06040042	The number and length of the objects to be mapped might exceed the length of the PDO.
06060000	Access failed due to a hardware error.
06070010	The data type is not correct or the parameter length is not correct.
06070012	Incorrect data type: the parameter is too long.
06070013	Incorrect data type: the parameter is too short.
06090011	The subindex does not exist.
06090030	The interval of values for the parameter is too large (only for write access).
08000000	General error.
08000020	The data item cannot be transferred or stored in the application.
08000022	The data item cannot be transferred or stored in the application due to the current state of the device.

9 Dip-switch configuration

Using the Dip-switches present on the motor it is possible to set the baudrate and the motor node number.

Dip-switch	Meaning
1	If bit 6 is set to On these indicate the baudrate, otherwise the node number.
2	
3	
4	
5	
6	If this is On it is possible to set the baudrate using bits 1 to 5. If it is Off, bits 1 to 5 represent the motor address.
7	When they are both set to On they insert the line termination. When both set to Off they cannot insert any termination. The other possible configurations should be avoided.
8	

9.1 Baudrate

To set the required baudrate, turn the motor on with bit 6 on and bits 1 to 5 set according to the baudrate required:

Dip-switch					Baudrate	Confirmation blink
5	**4**	**3**	**2**	**1**		
Off	Off	Off	Off	**On**	1 Mbaud	1
Off	Off	Off	**On**	Off	800 Kbaud	2
Off	Off	Off	**On**	**On**	500 Kbaud	3
Off	Off	**On**	Off	Off	250 Kbaud	4
Off	Off	**On**	Off	**On**	125 Kbaud	5
Off	Off	**On**	**On**	Off	100 Kbaud	6
Off	Off	**On**	**On**	**On**	20 Kbaud	7

Immediately after starting, the green motor LED will light up for approximately one second. After this, if the baudrate value desired has been successfully saved to the flash memory, the LED will start to flash periodically with the number of blinks for the chosen baudrate. If the Dip-switch configuration is not among the ones listed above, the baudrate will remain the one set previously. The motor is supplied with an initial baudrate of 500 Kbaud.
If there have been problems saving to the flash memory, the green LED will start to flash in intervals of 4 seconds (2 seconds on and 2 seconds off).
At the end of the procedure, turn the motor off, lower Dip-switch 6 and set the required node number on Dip-switches 1 to 5.

9.2 Restoring Default parameters

When Dip-switches 5 and 4 are set to On it is possible to restore all the motor's default parameters, including the baudrate. At the end of the procedure the LED will flash with the code indicated in the previous paragraph "Baudrate". This function is present starting from version 123 of the firmware.

9.3 Node number

If the motor is turned on with Dip-switch 6 set to Off, the Smart Motor starts normally, reading the node number from Dip-switches 1 to 5.

:

Dip-switch					Node number
5	**4**	**3**	**2**	**1**	
Off	Off	Off	Off	Off	0
Off	Off	Off	Off	**On**	1
Off	Off	Off	**On**	Off	2
…	…	…	…	…	…
On	**On**	**On**	**On**	Off	30
On	**On**	**On**	**On**	**On**	31

The node number may vary from 0 to 31, and once the motor has been turned on it can no longer be modified. In other words, to change the node number it is necessary to turn the motor off and then on again.

9.4 Meaning of the green LED

The flashes emitted by the Smart Motor green LED have a precise meaning during the procedure to assign the baudrate described in the paragraph Baudrate (the meaning is also described in this paragraph). On the other hand, if the motor is turned on with Dip-switch 6 set to Off (normal operation) you will have the following:

- The LED flashes at half-second intervals (250 milliseconds on, 250 milliseconds off): the motor is operating correctly and is in pre-Operational state as described in "DS301 States".
- The LED is permanently on: the motor is operating correctly and is in Operational state.
- The LED flashes at 200 millisecond intervals (100 milliseconds on and 100 milliseconds off): during the initial autocheck phase the motor has detected that the firmware saved in the flash memory is not correct. In this case the motor is not operational and it will be necessary to set the firmware again. Please contact the supplier.
- The LED flashes at 64 msec intervals (32 msec on and 32 msec off): internal error. One of the causes may be that incorrect PDO mapping parameters have been saved to the flash memory (too long or wrong length, non-existent object to be mapped,...). If necessary, contact the supplier.